디자인학 박사 학위논문

컴퓨터 프로그래밍 기법을 사용한 키네틱 한자 의미 표현 연구

Research on the Expression of Meaning of Kinetic Chinese Characters using Computer Programming Techniques

2020년 2월

서울대학교 대학원

디자인학부 디자인 전공

리 신 린

컴퓨터 프로그래밍 기법을 사용한 키네틱 한자 의미 표현 연구

지도교수 김 수 정

이 논문을 디자인학 박사학위논문으로 제출함
2020 년 2 월

서울대학교 대학원
디자인학부 디자인 전공
리 신 린

리신린의 박사 학위논문을 인준함
2020 년 2 월

위 원 장 김경선 (인)

부위원장 이장섭 (인)

위 원 김수정 (인)

위 원 채정우 (인)

위 원 이병학 (인)

국문초록

과학기술 문명은 사회의 모든 분야를 바꾸어 놓았을 뿐만 아니라 디자인 분야에도 커다란 변화를 가져왔다. 1980년대 이후의 젊은 디자이너들은 태어나자마자 디지털 세계에 포위되었다고 말할 수 있을 정도로 과학기술의 영향을 받으며 자라왔다. 이들은 다양한 과학기술을 적극적으로 예술에 활용하였고 과학기술은 예술가들의 작품 활동에 지대한 영향을 미쳤다. 현대 예술가들은 이제 물리 세계의 재료와 도구를 작품에 활용하는 구상 방법 외에도 가상의 스크린 공간 위에 펼쳐질 2차원, 3차원 심지어 4차원적인 디지털 아트 표현 기법에 대해서도 탐색하는 노력을 펼칠 필요가 있다. 현대미술과 과학기술의 융합이 날로 확장되면서 현대적인 디지털기술을 능숙하게 다루는 능력(technic)은 차세대 디자이너들이 새로운 산업시대에 살아남는 경쟁력이 되고 있다.

서양의 디자이너들은 아주 이른 시기부터 기술과 디자인을 접목하는 노력을 펼쳐왔고 이를 통해 체계적인 연구 방법들을 도출해낼 수 있었다. 디지털 기술의 발전에 비해 동양의 연구와 도입 시기는 비교적 늦은 편이다. '어떻게 이러한 이론들을 동양에 뿌리내리게 하고 꽃피우게 할 것인가?', 이는 전 아시아 디자이너들이 당면한 공통적인 과제라 말할 수 있다. 아시아 예술가의 일원으로서 본인은 한자의 가능성에 주목하며 컴퓨터 코딩 기술과 전통 한자 디자인을 접목하는 기법을 시도해 보았다.

"문자"는 인류 문명의 정수이자 디자인 분야에서 주로 연구하는 대상이기도 하다. 손수 쓰는 필기체이건 디지털 타이포그래피이건 "문자 디자인"은 줄곧 기술의 발전과 긴밀하게 맥을 같이 해왔다. 그러나 종이를 매개로 하는 기존의 폰트 디자인 기법은 스크린 위의 "동적인(kinetic)" 공간에서 점차 적응을 하지 못하고 밀려나기 시작했다. 이와 더불어 컴퓨터, TV, 휴대 전화 또는 일부 스크린 매체에서 간단하게 바꾸는 방식도 "문자가 움직이는" 키네틱 디자인 문제를 본질적으로 해결할 수 없었다. 이에 현재 일부 디자이너들이 다시 "문자" 자체에 주목하며 자국 문화의 토대 위에서 자국 문자의 속성과 역사 등을 디지털 매체의 접목시키기 위해 다양하게 연구하고 있다.

이에 본인도 다음과 같은 문제를 고찰하게 되었다. '디지털 시대 우리

는 어떠한 방식으로 문자에 움직이는 속성을 가해야 하는가? 한(漢) 나라 문화를 계승하고 있는 한자는 또 어떠한 방향으로 나아가야 하는가?' 본인은 위의 내용을 담아 논문을 총 6장으로 나누어 "키네틱 한자"에 대한 연구를 진행하였다. 연구 내용 및 방법은 대략적으로 다음과 같다.

가장 먼저 키네틱, 키네틱 아트, 키네틱 타이포그래피의 정의와 역사적인 맥락을 정리하였다. 선행 연구 자료와 관련 문헌을 통해 키네틱의 과거와 현재 그리고 미래를 조망해보고 키네틱 관련 개발 기법과 개발 도구에 대해 구체적인 분석을 진행하였다. 본고에서 제시한 개인 작품은 모두 프로세싱(Processing) 언어를 사용하여 창작한 것들이다.

그다음으로 한자의 특성과 발전 방향에 대한 연구를 진행하였다. 키네틱 타이포그래피의 특성을 종합하여 키네틱 한자의 디자인적 특징을 도출했다. 이와 같은 연구를 통해 다음과 같은 사실을 명확히 알 수 있었다. 키네틱 한자를 창작하는 과정은 한자가 시대의 변화에 적응하기 위해 꼭 필요한 것이며 또한 문자 자체의 특성을 유지하는 것이기도 하다. 논문에서 작품 사례를 분석한 부분에는 다른 작가의 디자인의 작품 외에도 본인이 직접 참여한 작품도 포함되어 있다. 해당 작품들을 비교 대조하면서 얻은 작품 개발 노하우는 이후의 창작 방향과 구상을 더 명확하게 만들어주었다.

마지막으로 한자의 구성 요소를 바탕으로 16개의 작품을 창작하였다. 작품들의 주제는 각기 다르며 순수하게 컴퓨터 코딩과 알고리즘만을 사용해 만든 것들이다. 그리고 매 작품마다 작품 감상 에세이를 덧붙였다. 이는 서양에서 종종 작품 감상 에세이를 통해 관객과 상호 소통하는 문화와 더불어 그림과 시를 함께 제시하는 아시아 전통 예술의 "시화동원(詩畵同源)"이라는 방식에서 비롯되었다.

논문의 연구를 통해 타입(type)과 타이포그래피(Typography)에 코드(code)를 결합하니 본래 고정 불변하던 시간축(timeline)이 달라지는 것을 엿볼 수 있었다. 제너레이티브 (generative) 기술이 더해지면서 작품의 콘텐츠는 더 풍부해지고 예측 불가능하게 바뀌었다. 이로써 키네틱 타이포그래피의 창작 기법을 한자에 적용할 수 있고 한자의 특성과 결합한 키네틱 디자인은 무궁한 잠재력을 갖게 된다.

“키네틱 한자” 디자인 분야는 현재 부단히 확장되고 있지만 모호한 양상과 더불어 아직 명확하게 정의 내려지지 않은 상태를 보이고 있다. 그러나 우리는 유구한 동방 문화와 풍부한 심미관(審美觀)을 가지고 있다. 이를 바탕으로 새로운 기술과 새로운 디자인을 접목하는 노력을 더 한다면 “키네틱 한자” 디자인은 향후 디자인의 발전을 이끌어 가는 새로운 트렌드가 될것이 분명하다.

주요어: 키네틱. 키네틱 아트, 한자, 프로그래밍, 타이포그래피
학　번: 2017-37780
E-mail: ***

목 차

표 목 차

그 림 목 차

1. 서론

1.1 연구 배경 및 목적

1.2 연구 문제 및 과정

1.3 연구 방법 및 범위

제1장 서론

1.1 연구 배경 및 목적

20세기 예술가들 사이에서 “안료를 사용하는 전통 회화 양식”에 대해 의문을 제기하는 비판적 시각이 있었다. 20세기 초에 조르주 브라크(Georges Braque)와 파블로 피카소 (Pablo Picasso) 는 신문지, 테이블보 천 조각, 노끈 등 일상의 소재를 작품 담아내는 창작 세계관을 통해 캔버스 위의 안료 종류를 다양하게 확장하고 참신함을 구가하였다. 이 “캔버스 위의 투혼”은 20세기 다양한 예술가들에 많은 영감을 불러일으켰다. 러시아 화가 카슈미르 말레비치[1] (Kazimir Malevich) 와 블라디미르 타틀린[2] (Vladimir Tatlin) 에서부터 20세기 중반의 잭슨 폴록 (Jackson Pollock) 그리고 리처드 프린스 (Richard Prince) 에 이르기까지 그들은 모두 전통 회화 양식에 대해 회의적인 반응을 보이며 이 같은 대열에 동참했다. 이들 외에도 익히 모두가 알고 있는 마르셀 뒤샹(Marcel Duchamp)이 또한 새로운 방법을 시도하며 아티스트를 예술 작업의 중심에 두었고, 이러한 노력에 힘입어 아티스트는 더 이상 “캔버스”의 제약을 받지 않으며 자유로운 방식으로 다양한 예술을 표현할 수 있게 되었다.

표현 매개체에 대한 아티스트들의 이 같은 생각은 사실 산업 혁명을 겪으면서 발생할 수밖에 없는 필연적 결과이다. 20세기의 전위 예술 아방가르드(Avant-garde) 와 동일하게 이러한 움직임은 “기술 혁명” 속에서 지속성을 보이며 하나의 유파를 형성한다. 해당 유파는 주류 예술에서 벗어난 생경한 분야에서 먼저 시작된다. 이들은 건축가와 엔지니어들

---

1) Kazimir Malevich(1879~1935), 러시아의 화가, 교사, 이론가. 절대주의의 창안자로 국제 추상주의 운동의 중심적인 역할 수행.

2) Vladimir Tatlin(1885~1953), 카지미르 말레비치와 당대의 다른 예술가들과 함께, 1920년대 소비에트 연방의 러시아 아방가르드 및 러시아 구성주의의 대표적인 작가의 한 명으로 다루어지다.

이 주류를 이루어 활동하는 과학 기술 영역을 과감하게 예술 분야에 도입한다. 이 같은 행보는 키네틱 아트의 선구자 나움 가보(Naum Gabo)와 나즐로 모홀리 나기 (Laszlo Moholy Nagy) 등의 인물을 연상시킨다. 그들은 “기계”를 예술 창작의 “협력 파트너”로 바라보았다. “기계”는 20세기 후반에 본격적으로 예술 창작에 활용되면서 완전히 다른 성격을 띠게 된다. 과거 몇 세기 동안 “기계”는 물리 운동을 하는 기기로써 줄곧 인간의 근육을 대체하는 의미로 연관되어 사용되거나 또는 인류의 기타 유기적 특징으로 연관되어 사용되어왔다. 그런데 단순하고 부수적인 기능에 불과했던 기계 시대가 막을 내리며 “기계”는 두뇌의 인지 기능을 모방하거나 본뜬 것으로 설계되었고, 컴퓨터와 같은 전자 기계들이 물리 운동을 하는 작업 기기들을 빠르게 대체하면서 디지털 기기는 인류에 중대한 영향을 끼치기 시작한다.

날로 발전하는 시대의 변화에 발맞추기 위해 컴퓨터 분야를 모색하고 기술과 예술을 융합하는 창작 작업은 당대 아티스트와 디자이너들이 필연적으로 걸어야 할 길이 되고 있다. 과거 우리는 붓과 안료를 사용하여 대뇌와 눈을 통해 보이는 사물을 그리고 양손으로 직접 예술적 구상을 다듬어가며 도화지나 캔버스 위에 시각적 이미지들을 담아냈다. 이러한 고전적인 수작업 방식을 통해 많은 예술품들이 만들어졌다. 이후 기계 시대로 진입하며 기계 장비의 활용이 가능해지면서 예술가들은 필름이나 다른 매개체를 이용하여 정보를 받아들이기 시작하였고 한층 더 사실감 있는 이미지를 출력하는 방식으로 새로운 예술 작품을 제작하였다.

현대의 디지털 시대에는 다수의 아티스트가 컴퓨터와 같은 첨단 장비를 예술 창작에 적극적으로 활용하고 있다. 수학 언어를 컴퓨터 전산 시스템에 입력한 다음 스크린을 통해 참신한 이미지를 출력하는 방식으로 디지털 융합 예술작품을 창작한다. 컴퓨터 아트는 1970년대 초반에 역량이 집중되기 시작하였는데, 80년대에 이르러 컴퓨터가 더욱더 값싸고 편리해지면서 아티스트 (새로운 매개체를 사용하여 작품 창작 활동을 하는 아티스트) 들이 이를 대대적으로 작품 창작에 응용하였다. 이와 더불어

개인 PC의 대중화도 새로운 예술 장르의 등장에 상당한 기여를 했다. 개인 PC의 활용도가 날로 증가면서 컴퓨터를 기반으로 하는 컴퓨터 그래픽스, 애니메이션, 디지털 조각, 키네틱 아트, 생성 예술(Generative Art) 및 다양한 형태의 인터랙티브(interactive) 예술 등이 등장하며 디지털 예술은 발전을 거듭하고 있다.

한편 기술은 예술뿐만 아니라 디자인 분야에도 새로운 활기를 불어넣었다. 21세기로 들어서며 시각디자인은 단순하게 인쇄 매체를 활용하는 데에서 확장되어 스크린 매체를 기반으로 하는 멀티미디어와 융합하는 시대로 진입하였다. 이로써 정보 전달의 주요 매체인 “문자”도 다른 매개체의 영향을 받게 된다. 스크린에 재생되는 문자는 대부분 “전통 인쇄 조판(tradition of print typography)”에 기반하고 있지만 점점 더 많은 문자가 인쇄 이외의 다양한 환경에 놓이면서 문자 디자이너들도 자연스럽게 “신기술”과 접목하는 새로운 창작 방식이 필요하게 되었다.

문자가 새로운 매체와 결합하면서 “문자 디자인”은 정지 상태의 평면 디자인에서 보다 다차원적으로 움직이는 인터랙티브한 매개체로 새롭게 탈바꿈한다. 입체적으로 움직이는 형태의 사물은 보는 사람에게 더 쉽게 기억되어 관람객의 참여도와 체험감을 높이게 마련이다. 문학 학자 테무 이코넨[3](Teemu Ikonen)은 디지털 문헌과 프린트 문헌을 나누는 결정적 요소 중 하나가 바로 “텍스트의 움직임”이라고 말하였다. 문자의 움직이는 형태를 완성하는 과정에서 컴퓨터와 도형, 이미지 디자인 소프트웨어의 대중화가 이루어졌고 이로써 점점 더 많은 디자이너들이 해당 소프트웨어를 통해 디자인에 참여하게 되었다. 현재 대부분의 애니메이션과 움직이는 타이포그래피 창작은 모두 Adobe Flash와 After Effects와 같은 소프트웨어를 통해 만들어진다. 저렴한 비용과 간단한 조작으로 “움직이는 문자"는 이제 흔히 볼 수 있는 형태가 되었다. 멀티미디어의 역할은 기계적인 반복 운동에만 그치지 않는다. 멀티미디어를 통해 디지털 스크

3) Teemu Ikonen .《Moving text in avant-garde poetry -Towards a poetics of textual motion》, 2003, Cap 3.

린을 기반으로 하는 매체에서 “3D 타이포그래피 환경”을 구현해 내면서 보다 더 정교하게 “공간과 시간의 차원”을 표현하는 것도 가능해졌다. 또한 관련 소프트웨어를 능숙하게 사용하게 되면서 디자이너들은 20세기의 아방가르드 유파와 마찬가지로 끊임없이 창작의 새로운 돌파구를 찾고 있다. 최근에는 데이터 시각화(Data visualization), 파라메트릭 디자인(Parametric Design), 알고리즘 디자인(Algorithm Design) 등이 광범위하게 활용되면서 프로그래밍 사고와 디자인을 융합하는 방식이 디자이너들이 새롭게 모색하는 예술 장르가 되고 있다.

“키네틱 타이포그래피 디자인”은 유럽과 미국에서 빠르게 발전하며 서양 디자이너들의 중점 연구, 창작 대상이 되었고 다수의 우수한 디자인 작품들로 탄생했다. 다만 문화적 차이로 인해 서양 디자이너들은 아직 “키네틱 한자” 디자인 분야는 섭렵하지 못했다.

한자 문화는 동으로는 한국과 일본으로, 남으로는 베트남과 미얀마, 서로는 인도, 북으로는 몽골로 매우 광범위하게 전파되었다. 최근 아시아가 부단히 발전하면서 한자의 세계적인 영향력도 날로 커졌으며 많은 사람들이 한자와 관련된 다채로운 디자인 작품들을 기대하고 있다. 하지만 한자 자체가 가지고 있는 내용 이해의 어려움과 거대한 문자의 양 그리고 복잡한 필획과 체계화되지 못한 디자인 이론이 더해져 관련 기술이 뒤처지지 않음에도 불구하고 키네틱 타이포그래피에 대한 연구는 서양에 비교해 현저하게 부족하다. 게다가 전통 한자의 특징과 결합한 키네틱 연구는 더더욱 부족한 실정이다.

한자 디자인 연구는 주로 동북아 지역에 집중되어 있는데 그중에서도 한중일 지역의 연구가 가장 우수한 편이다. 현재 중국 한자 폰트 디자인 서적과 교재 등 자료들은 그 내용이 주로 평면 시각적인 캘리그래피, 인쇄 폰트 및 폰트 활용과 조판에 집중되어 있어 한자 키네틱 폰트 디자인에 관한 내용은 매우 부족한 상황이라고 말할 수 있다. 한국과 일본은 한자 디자인의 연구와 응용에 있어 기여도가 높으나 양국 연구도 움직이지 않는 정지 상태의 시각적 표현과 전달에만 머물러 있어 아쉬움이 남

는다.

이에 본 논문을 통해 키네틱 한자와 관련된 용어 정의를 정리하고, 키네틱 타이포그래피의 디자인 방법과 형식에 관한 총괄적인 결과물을 한자 본연의 특징과 접목하여 향후 대대적으로 발전하고 응용될 키네틱 한자 디자인에 관한 여러 가능성을 제시하고자 한다. 더불어 컴퓨터를 정보 전달의 매개체로 삼아 프로그래밍 방식을 키네틱 한자의 작품 창작 활동에 응용하여 한자가 내포하고 있는 의미를 효과적으로 표현하고, 동시에 시각적인 효과를 한층 더 현대적인 감각으로 만들어내고자 한다. 최종적으로는 한자 키네틱 디자인의 합리적인 방안을 제시하여 한자 디자인을 뉴미디어 환경에 적응하게 하는 동시에 키네틱 한자 디자인의 새로운 지평을 열고자 한다.

1.2 연구 문제 및 과정

본 연구자는 “한자”의 키네틱 디자인을 주제로 한자의 키네틱 타이포그래피적 가치를 탐구하고 키네틱 한자의 구성적 특징과 한자 폰트의 키네틱 표현 방법, 한자의 키네틱 폰트 배치 방법을 고찰하여 최종적으로 컴퓨터 프로그래밍을 응용해서 한자가 내포하고 있는 의미를 표현하는 방법에 대해 논하고자 한다. 세부적 연구 내용은 다음과 같다.

첫째, “키네틱” 관련 용어의 선행연구를 이론적 토대로 삼아 전반적인 경향과 디자인 요소를 분석하고 연구 범위와 연구 방법을 제시한다.

둘째, 기존 키네틱 디자인과의 차이점이 무엇인지 살펴보고 키네틱 타이포그래피의 분석 결과를 한자에 적용하여 키네틱 한자의 특징에 대한 분류 작업을 진행한다.

셋째, 본 연구자의 초기 작품에 대한 분석 및 대조 작업을 통해 키네틱 타이포그래피의 결과물을 종합한다. 해당 결과물은 최종 작품 창작의 방

향을 제시해준다.

마지막으로 이론을 실제 작품 창작에 응용하며 한자 키네틱 디자인의 방향성을 논의한다. 고정 관념과 상투적인 디자인 기법에서 탈피하여 새롭게 창작하는 한자 키네틱 디자인 기법, 기술적 수단을 활용해 한자 문자가 내포하고 있는 의미를 더 잘 표현할 수 있는 방법, 기존 콘텐츠를 더 현대적으로 표현할 수 방법 등을 모색한다.

논문은 총 6장으로 구성된다.
제1장에서는 키네틱 한자의 연구 배경 및 목적, 연구 문제 및 과정, 연구 방법 및 범위 등 내용을 제시하고 연구 방향을 모색한다.

제2장에서는 키네틱의 배경과 개념, 특징, 분류 기준을 자세히 기술하고 키네틱 아트의 발전 역사를 정리하여 작품과 관련된 컴퓨터 기법에 대한 이론적 고찰을 진행한다.

제3장에서는 키네틱 타이포그래피의 역사, 디자인 특징, 스타일, 가치와 효과를 서술하고 한자 디자인과 뉴미디어에 대해 다각도로 살펴본 다음 최종적으로 키네틱 한자의 디자인적 특징을 도출한다.

제4장에서는 본 연구자의 초기 실험작을 소개하고, 그중에서 디지털 컴퓨터 프로그래밍 기법을 활용하여 움직이는 형태로 만든 작품을 사례로 들어 이에 대한 분석을 진행한다.

제5장은 개인 작품의 창작 부분으로 작품의 배경과 기술적인 부분을 소개한다. 먼저 앞에서 도출한 키네틱 한자 디자인의 특징을 이론적 토대로 삼아 작품을 유형별로 분류하고, 그리고 나서 디자인 특징을 종합한 다음 한자 고유의 “선(線)”을 콘셉트로 한 작품을 최종적으로 완성한다.

마지막으로 제6장에서는 연구에 대한 요약과 함께 연구 결과 및 결론, 시사점을 정리하고 연구의 한계성 및 향후 연구 방향을 제시하고 논한다.

전체적인 연구의 설계도는 아래 [그림 1]과 같다.

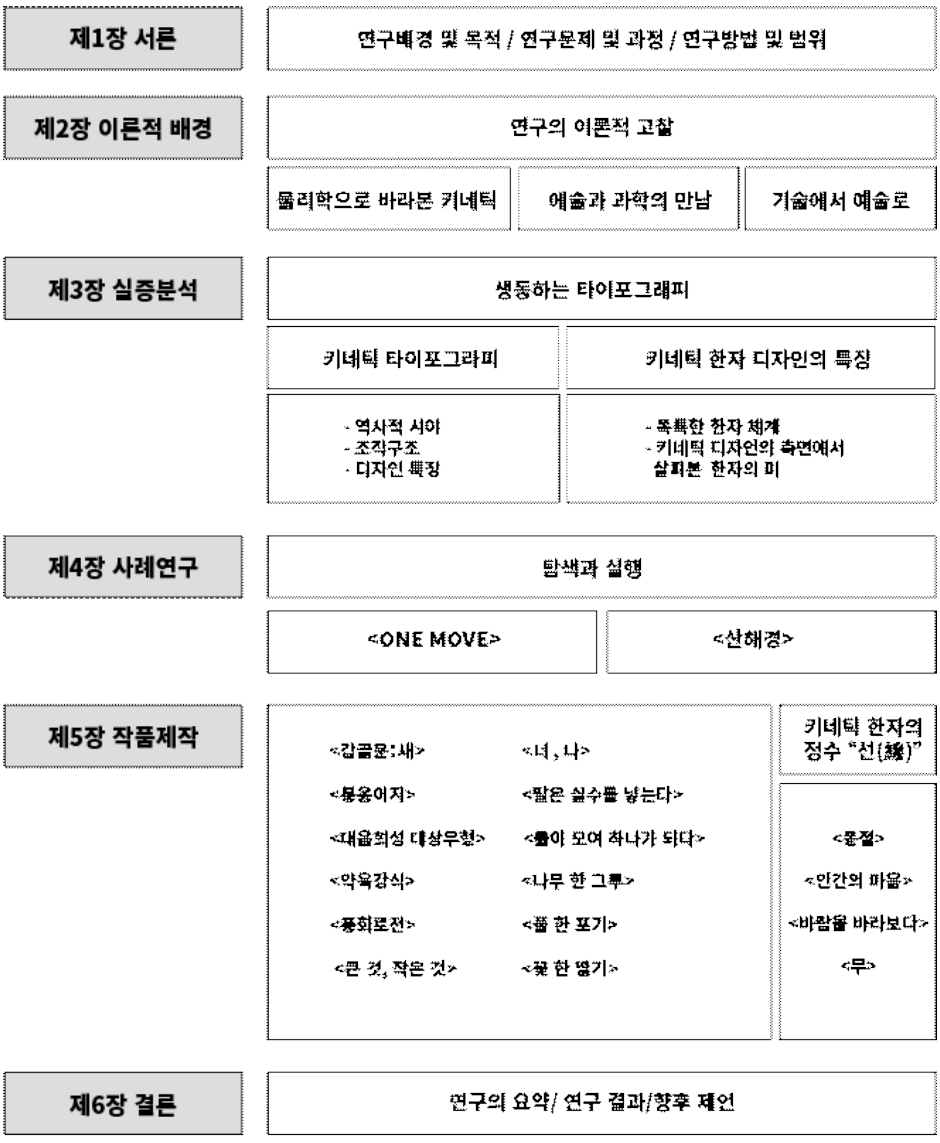


Figure 1. 연구의 설계도

1.3 연구 방법 및 범위

본문에서는 연구목적 및 연구 문제의 제기에 따라 문헌연구, 비교 연구, 사례연구, 귀납법 통해 기초연구, 응용연구 (Applied Research) 의 성과와 지식을 종합하여 작품 창작에 대한 연구를 실시한다. 키네틱 한자 디자인에 대한 이론적 측면과 기술적 측면을 탐색하여 보충하는 부분의 세부적인 분석 방법과 연구 방법은 다음과 같다.

1. 간행된 논문, 서적, 연구 보고서, 학술지 등 선행 문헌자료를 통해 키네틱의 기원과 개념과 범위, 특징에 대해 살펴보고 키네틱 아트의 발전 동향 등을 분석한다.
2. 표의 문자인 한자는 형태, 소리, 의미로 구성되어 있는데 이 3요소는 한자의 시각적 이미지, 구조 및 디자인과 밀접하게 연관되어 있다. 이에 국내외 키네틱 타이포그래피의 정의와 디자인 수법을 종합하여 키네틱 한자의 창작 형식과 방법을 분석한다.
3. 본 연구자의 창작 작품과 연관 지어 키네틱 한자 디자인의 현대와 전통적 차이 및 융합에 대해 살펴보고 어원학, 시지각적 이론, 기호학, 그래픽스, 전통 미학과 현대 미학에 대한 연구를 통해 얻은 결과물을 최종 작품 창작에 응용한다.

연구 방법의 하나로, 본 연구자는 컴퓨터 프로그래밍을 사용하여 움직이는 현상을 담아냈다. 프로그래밍 안에는 기초 물리학, 기하학 및 알고리즘이 포함된다. openFramework, Paper.js, Unity, P5 등 프로그램에 정통하지 않고 vvvv, NODEBOX, Grasshopper의 이미지 표현에 있어서도 제한적이어서 본인 주로 processing을 작품 제작의 도구로 삼았다.

“움직이는 폰트”와 “폰트의 움직임” 이 둘의 차이점은 움직이는 형태로 폰트를 표현하는 데 있는데 전자가 바로 이에 속한다. 전자는 폰트 디자인의 한 종류이고, 후자는 폰트 디자인이 아닌 기존의 폰트에 움직이는 배열을 조합하는 것을 지칭한다. 한자는 형태, 소리, 의미가 하나의

문자로 구성되는 특징을 가지고 있다. 이를 감안하여 한자 키네틱 디자인 시 타입 페이스, 타이포그래피를 모두 디자인 범주 안에 포함시켰다.

이에 따라 본 논문의 연구 범위를 두 가지로 정하였다. 하나는 키네틱 기법을 사용하여 한자 폰트를 표현하는 한자 폰트 자체에 대한 연구이고, 다른 하나는 한자 폰트를 사용하여 진행하는 키네틱 편집 배열(한자 폰트 자체에서 벗어나 글자와 글자 사이의 관계를 연구함)로 한정하였다. 위의 내용을 반영한 최종 작품은 모두 컴퓨터 기법을 활용하여 키네틱 한자의 디자인적 특징을 담아 재창작한 작품들이다.

2. 이론적 배경

2.1 물리학으로 바라본 키네틱

2.2 예술과 과학의 만남

2.3 기술에서 예술로

제2장 이론적 배경

2.1 물리학으로 바라본 키네틱

Dr.Leonard Shlain는《ART&PHYSICS》에서 예술가와 물리학자의 관계에 대해 묘사하며, 이 둘의 공통점은 바로 각자의 방식으로 자연계 이면의 법칙을 이해하려는 데에 있으며 궁극적인 목적도 서로 동일하다고 바라보았다. 물리학은 가장 오래된 학문 중 하나로 영문명은 physics이고 자연을 뜻하는 그리스어 Φύσις에서 유래했다. [4] 만약 우리가 모종의 자연 현상을 이해하고 또 그것을 활용하여 인류와 지구를 더 좋게 바꿀 수 있다면 이것은 곧 우리가 인류 전체 지식에 대해 기여한 것이라고 말할 수 있다. 이러한 사고방식을 예술 분야에 적용하여 물리학 지식을 이해하고 응용함으로써 우리는 새로운 예술적 영감을 불러일으킬 수 있다.

물리는 자연과학 학문으로 갈릴레이와 뉴턴의 시대에서 시작되었다. 3세기가 넘는 발전을 거듭하며 현재에는 다수의 분과로 나뉘었지만 물리학은 여전히 사람들의 존경과 사랑을 받고 있는 기초 학문이다. 사실 물리학의 원리는 모두 '힘(力)'을 핵심 키워드로 하는 연역적 추론에서 비롯된다. "역학(力學) Mechanics" 은 물리학의 한 분야로, 외력을 받고 있는 물체의 정지 또는 운동 상태를 설명하고 예측하는 자연 과학이다. [5] 역학은 정역학(精力學, Statics)과 동역학(動力學, Dynamics) 두 갈래로 나뉘며, Dynamics은 다시 운동학(運動學, kinematics)과 동역학(動力學, Kinetics) 두 가지로 나뉜다. 브리태니커 백과사전 (ENCYCLOPAEDIA BRITANNICA) 에 Kinetics은 다음과 같이 기술되어 있다 :

Authors using the term kinetics apply the nearly synonymous name dynamics (q.v.) to the classical mechanics of moving bodies. [6]

최근에는 다수의 교과서에서 내용의 혼동을 막기 위해 점차 Dynamics

---

4) 나무위키, https://namu.wiki/w/물리학
5) 위키백과, https://ko.wikipedia.org/wiki/역학
6) 브리태니커 백과사전. https://www.britannica.com/science/kinetics

을 kinematics과 Kinetics으로 대체하여 사용하고 있다. 물리학의 구조는 아래의 그림과 같다.

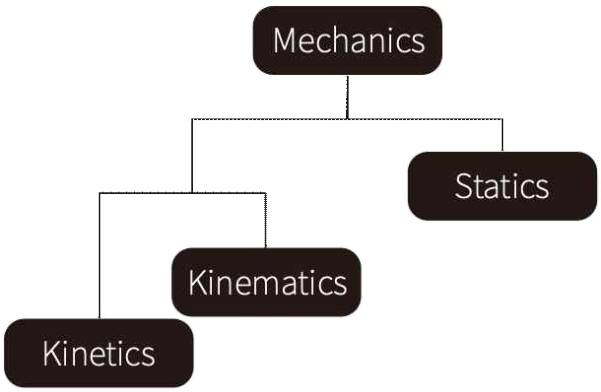


Figure 2. 관계 구성도

정역학(靜力學, statics)은 정적 평형 상태에 있는 계를 다루는 분야이다. 즉, 이러한 상태에서는 하위 계들의 상대적 위치가 시간에 따라 변화하지 않으며, 구성 물질과 구조가 외부력의 작용 하에서 정지 상태에 있게 된다. 정적 평형 상태에서 계는 정지하여 있거나, 그 질량 중심이 일정한 속도로 움직인다.[7] 정역학은 물체가 정지한 상태에서 힘을 받는 상황을 다루기 때문에 본 논문에서는 이를 중점적으로 논하지 않는다.

운동학(運動學, Kinematics)은 입자나 물체 또는 다수의 물체가 모여 이루어진 계의 운동을 다루는 고전 역학의 하위 학문이다. 운동학은 운동의 양상을 다루지 운동이 일어나는 원인에 대해서는 고려하지 않는다.[8] 운동학은 운동 기하라고도 불리는데 운동의 기하적인 측면에서 접근하고 있기 때문이다. 그 외에도 운동학은 동역학의 이론적 토대를 제공한다.

동역학(動力學, Kinetics)은 물체의 “운동 상태”와 그것이 받는 “힘”의 관계에 대해 연구하는 학문이다. 동역학은 운동의 원인(力, force)을 연구

7) 위키백과 ,https://ko.wikipedia.org/wiki/정역학
8) 위키백과 ,https://ko.wikipedia.org/wiki/운동학

하는데 이것이 Kinematics와 Kinetics 간의 핵심적인 차이를 나누는 요소이다. 뉴턴의 운동법칙은 동역학에서 광범위하게 사용된다. 위키피디아에서는 Kinetics와 Kinetic을 모두 동일하게 해석한다(Ancient Greek: κίνησις, lit. 'kinesis', movement or to move).[9] 이 둘의 차이점를 살펴보면 Kinetics은 명사(noun)인데 반해 "Kinetic"은 형용사로써 운동의 움직임을 표현하는 데 있다. 또한 Kinetic은 일반적으로 조합되어 사용되는데 Kinetic theory, Kinetic energy, Kinetic art, Kinetic typography 등등을 그 예로 들 수 있다.

작품 창작할 때 우리는 일반적으로 차근차근 단계를 밟아가며 진행한다. 물리 지식을 학습할 때와 마찬가지로 말이다. 물리학을 학습할 때에는 가장 먼저 "정역학"을 배우고, 그다음으로 "운동학"을 연구하고 마지막으로"동역학"을 배운다. 그 이유는 바로 정역학이 물체가 정지 상태에 있을 때 힘을 받는 상태를 연구하는 학문이기 때문이다. 운동학은 물체가 힘을 받는 것을 연구하기보다는 오로지 운동 상태만 관심을 기울이고 연구한다. 이와는 달리 동역학은 이들 양자에 대한 결합을 진행한다. '힘(力)'과 '움직임(動)'사이의 관계를 연구하기에 우리가 이를 동역학이라 부르는 것이다. 한 물체가 움직이기 시작할 때 움직임의 빠름과 느림은 속도를 가하는 것과 관련되어 있는데 이것이 바로 동역학의 예이다.

본고에서 언급하는 작품은 모두 컴퓨터 프로그래밍과 관련되어 있다. 작품에 컴퓨터 언어를 사용하여 키네틱 '움직임(動態)' 효과를 넣을 때는 보통 물리 상식과 수학 연산을 결합하여 진행한다. 물체를 가상 환경에서 더 자연스럽게 표현하기 위해 물체에 중력, 인력, 밀어내는 척력 Repels 등등의 외부에서 작용하는 힘인 외력(外力)을 더하는 것을 그 예로 들 수 있다. Kinetic art, Kinetic typography 등 형태의 용어가 등장하면서 많은 작가들이 작품 창작을 할 때에도 Kinetic 용어를 광범위하게 사용하고 있다. 위에서 상술한 내용을 살펴본 바와 같이 "Kinetic" 용어를 사용하는 것은 본 논문 최종 작품의 주제 의식과 일치한다.

---

9) 위키백과,https://en.wikipedia.org/wiki/Kinetic

2.2 예술과 과학의 만남

Kinetic Art는 한자로 동태(動態) 예술, 동력(動力) 예술 또는 움직이는 예술로 번역할 수 있다. “키네틱 아트” 용어는 ‘움직임’을 의미하는 ‘Kinesis’라는 그리스어에 그 어원을 두고 있는데, 움직임을 본질로 하는 미술을 지칭할 때 이 용어를 가장 많이 사용한다. 10)

최초의 키네틱 아트 작품은 바로 작가 마르셀 뒤샹(Marcel Duchamp)의 <자전거 바퀴 Roue de bicyclette>이다. 아래의 그림처럼 등받이 없는 나무 의자 위에 자전거 바퀴 하나가 거꾸로 놓여 있는데 이 작품이 바로 첫 번째로 공인된 키네틱 아트 제품이다.

Figure 3. Marcel Duchamp, Roue de bicyclette, 1913

10) 키네틱 타이포그래피는 키네틱 아트에서 가져온 용어로 키네틱 아트는 움직임을 주요 요소로 하는 예술 작업, 즉 작품 그자체가 움직이거나 또는 움직이는 부분이 조립된 것 등을 포함하는 개념이다. - 신청우, <디지털 타이포그래피> ,p150.2003.

1920년에는 구성주의 작가 나움 가보(Naum Gabo)와 앙투안 페브스너(Antoine Pevsner)가 리얼리스트 선언(The Realist Manifesto)에서 조형예술의 새로운 개념인 움직이는 리듬 (Rythmes cinétiques)을 제시한다. 그들은 물체 운동의 “시간” 요소를 예술 표현의 범주 안에 넣었다.

리얼리스트 선언의 제5 원리 내용은 다음과 같다. “우리는 정적인 리듬을 조각, 회화 예술의 유일한 요소로 삼았던 예술계의 천 년 동안의 망상을 거부한다. 우리는, 우리가 실제의 시간을 지각하는 기본 형식으로서 움직이는 리듬이 조각과 회화예술에 있어 새로운 요소임을 확인한다”. 같은 해, 나움 가보는 키네틱 조형 작품 “standing wave”를 완성하였다. 그는 독립적인 금속판을 사용하여 전기 모터로 움직임을 가해 공기 중에 미묘하게 물결치는 웨이브를 구현해냈다. 해당 작품은 운동 상태를 표현한 최초의 현대 예술 작품이다.

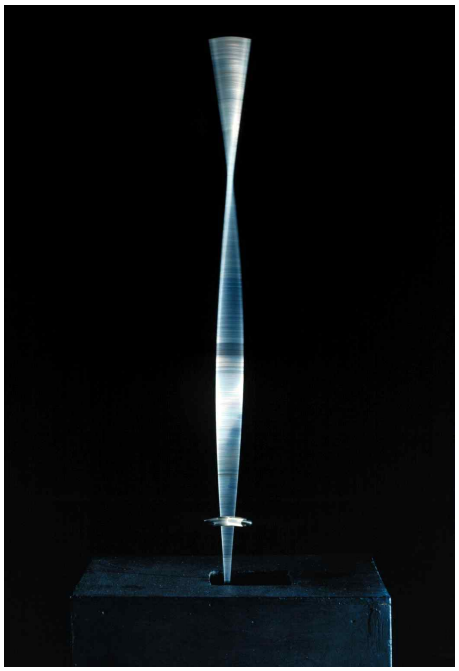


Figure 4. Naum Gabo . Kinetic Construction (Standing Wave) , 1919 - 20

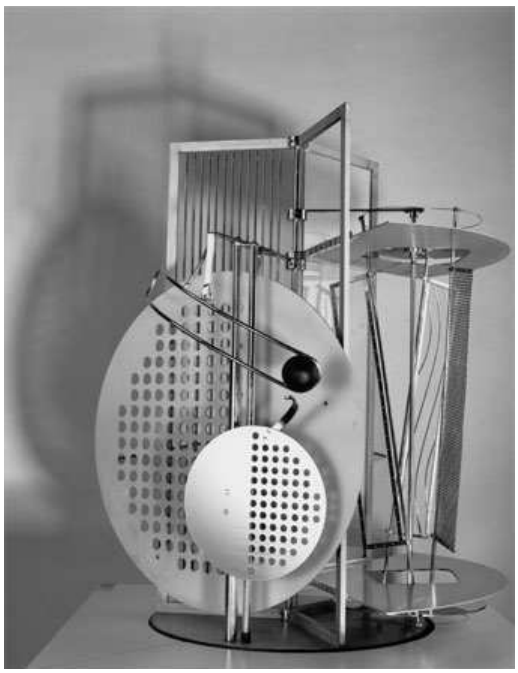


Figure 5. László Moholy-Nagy. Light Space Modulator,1930.

10년 후 바우하우스 작가 모홀리 나기(László Moholy-Nagy)가 또다시 “키네틱” 용어를 사용하여 그의 작품 ‘빛-공간 변조기 (Light-Space Modulator)’을 묘사한다. 그와 동시대에 살았던 후기구조주의(post-structuralism) 작가들이 1930~40년대 사이에 창작한 여러 작품들도 키네틱 예술이라고 불린다.

키네틱 아트가 하나의 집단적 경향으로 확립되기 시작한 것은 1955년 파리 드니즈 르네 (Denise René) 화랑에서 열린 “움직임”전과 그 선언문에서부터이며, 1965년 뉴욕 현대미술관의 “반응하는 눈 The Responsive Eye” 작품전, 1976년 파리 시립 현대미술관에서 열린 “빛과 움직임”전 등 많은 국제전이 미국과 유럽 각지에서 열렸다. 키네틱 아트는 하나의 작품 경향으로 인정되자마자 개인적 탐구의 단계로 들어서게 되어 1970년대 이후부터 현재까지 개인 작업을 통해 지속되고 있다.

조지 릭키(George Warren Rickey)는 1970년대 이전의 키네틱 아트 작품을 간략하게 분류하여 대략 아래의 6가지로 구분 지었다.[11]

1. ‘시지각적 현상 Optical phenomena’을 탐구하는 작품. 이러한 작품에서는 오브제의 시각적인 움직임 또는 실제적인 움직임, 때로는 관찰자의 움직임이 긴장감이나 놀라운 효과를 산출한다. 브리지트 라일리의 <흐름>(1964)이 그 예이다.

2. ‘변형 Transformations’현상을 응용하는 작품. 이러한 작품에서는, 움직이는 바퀴의 살이 사라지듯이, 빠른 움직임에 의해 대상은 비물질화되는 것으로 보인다. 또는, 아감 Yaacov Agam의 작품과 같이 대상이나 관람자의 움직임이 외관상의 풍부한 변화를 가져오기도 한다.

3. 움직일 수 있는 작품. 관객 자신이 변경시키거나 재구성할 수 있는 회화나 조각이 이에 속한다. 어떤 경우에는 공간이나 표면의 완전한 재구성에 이르기도 한다. 탈만 Talman 의 <Kugelbild>이 대표적인 예다.

---

11) 조지 릭키 [저] ; 윤난지 옮김. < 키네틱 아트 Kinetic art>. p27.

4. 다양한 형태의 '기계들'을 들 수 있다. 보통 전기 모터로 추진되며 왕복, 상하, 회전 운동을 하는 기어, 캠, 크랭크와 레버 등을 갖추고 있다. 반 그래브니츠 Gerhard van Graevenitz 의 기어 달린 회전체가 그 예이다.

5. '움직임에 근거한 빛의 작용'을 이용한 작품. 이러한 작품들이 이용하는 빛의 작용은 움직이는 복합적 표면에 투사된 광선이 만들어낸 그림자이거나 반영(半影)일 수도 있고, 스크린 위에서 리드미컬하게 움직이는 카메라로 찍은 일련의 형상들일 수도 있으며, 움직이는 빛의 원천에서 투사되어 실내의 어두운 벽이나 물체, 관람자 위에 일종의 우연한 밤하늘을 만들어내는 빛의 반점일 수도 있다. 줄리오 르 빠르끄(Julio LeParc)의 <연속적 광선의 가변성 Instabilite Continuelle Lumiere>이 그 예이다.

6. '움직임 자체'를 추구하는 작품. 이 경우는 보통 동력 수단을 절제하고 기계를 드러내지 않는다. 예로 Panayiotis Vassila- Kis Takis 의 10피트 높이의 흔들리는 용수철 막대기를 들 수 있다.

다만 위의 6가지 유형은 현대 과학기술을 집약하여 표현하는 작품 예를 들면 멀티미디어 기술과 시지각적 현상을 결합하거나, 컴퓨터 연산을 통해 구현하는 Generative 이미지 또는 하드웨어와 참여자가 상호 작용(interact) 하는 등의 새로운 작품이라고는 말할 수 없다. 하지만 이러한 초기 작품의 분석을 통해 우리는 키네틱 아트의 핵심 가치관에 대해 명확하게 이해할 수 있고 키네틱 아트의 발전 방향에 대해서도 판단할 수 있어 향후 작품 창작의 기반이 된다.

"움직임"의 미술도 하나의 용어로 정의하기에는 너무 다양하게 나타나고 있다. 이를 "움직임"의 특성에 따라 "환영적(illusionistic: 幻影的) 움직임"과 "실제의 움직임"으로 분류할 수 있다. 포펠은 이를 "가상의 움직임(mouvement virtuel)"과 "실제의 움직임(mouvement reel)"으로 분류하

였다. [12] 전자는 움직이는 것 같이 보이는 정지된 작품이고. 후자는 실제로 움직이는 작품이다. 어느 경우에나 평면적인 것일 수도 공간적인 것일 수도 있다.[13]

대체로 “Kinetic Art”는 “실제의 공간적 움직임”을 강조하는 용어로 사용되는데, “Cinetisme”, “I Art Ci-netique”, “Kinetische Kunst”등 유럽 용어에서는 “환영적이고 평면적인 움직임”까지도 포함한 넓은 의미로 사용되고 있다.[14] 즉, 움직임을 본질로 하는 미술은 실제로 움직이는 작품과, 실제로 움직이지는 않지만 시각적으로 움직이는 것같이 보이는 작품의 두 가지로 분류할 수 있다. 옵 아트(Op Art) 와 지각적 추상(Perceptual Abstraction)은 후자의 경향을 지칭하는 용어로서, 움직임의 시지각적 현상을 강조하는 용어이다. 키네틱 아트는 주로 실제로 움직이는 작품을 일컫지만, 옵 아트나 지각적 추상이라 불리는 경향까지도 포함하는 가장 포괄적이고 또한 보편적으로 통용되는 용어이다.

Kinetic Art가 다른 예술의 형식과 구별되는 중요한 특징은 “움직임”에 있다. 움직이는 효과를 구현하기 위해 예술가들은 반세기가 넘는 시간을 노력해야 했고 당시의 제한적인 기술 환경 속에서 창작을 진행했다. 오늘날 키네틱 아트는 현대 과학기술과 긴밀하게 연결되어 다양한 가능성을 창출한다. 본 논문은 그중에서도 멀티미디어의 환경에 놓여 있는 키네틱 디자인을 다루고 있다. 컴퓨터 프로그램을 창작 도구로 삼아 가상 운동 효과를 만들고, 스크린을 통해 그 가상의 결과물을 표현한다. 이에 따라 위에서 상술한 전통 키네틱 아트의 분류에 의거하여 “프로그래밍으로 구현한 가상 키네틱 예술”도 키네틱 아트 범주 안에 포함될 수 있다고 생각한다.

---

12) Frank Popper(1970), < l'art cinétique > ,p.275

13) 포펠은 실제의 움직임만 “공간적(spatial)”인 것과 “비-공간적(non-spatial)”인 것으로 분류하였다. P275.

14) 尹蘭芝, 움직이는미술에 관한 연구- 확장된 작품개념을 중심으로, 美術史學科, 박사논문 p3.

2.3 기술에서 예술로

1949년 펜실베이니아 대학에서 세계 최초로 전자식 컴퓨터 ENIAXC (Electronic Numerical Integrator and Calculator, 대형 창고 하나 정도의 크기)를 내놓았고 1951년에는 숫자와 텍스트 정보를 처리하는 상용 전자 컴퓨터가 특허를 받았다. 당시 정부가 지원하는 연구센터에서는 컴퓨터 기술에 대한 연구와 실험을 계속 강화하였지만 음악, 예술 등 분야와의 관련성은 매우 적었다. 과학자 출신인 연구원들이 예술에 대해서는 비전문가였기 때문이다. 이런 연유로 초기에는 “컴퓨터 아트”에 대한 미적 기준도 온전히 정립되지 않았었다.

현재에는 아티스트들의 대다수가 컴퓨터를 사용하고 있으며 디지털 기술로 직접 작품을 제작하고 있다. 컴퓨터는 우리 주변에서 손쉽게 만나는 ‘아티스트’가 되어 아티스트의 창의적인 발상을 실현하는 매개로써 그 역할을 톡톡히 하고 있다. 하지만 컴퓨터가 아티스트를 위해 전문적으로 만들어진 것은 아니어서 일부 아티스트는 Adobe Flash, Ps, AE 등 프로그램의 한계를 느끼고 표현 도구의 제약에서 벗어나 다채로운 예술을 표현하기 위해 보다 심층적인 컴퓨터 구조 원리와 프로그래밍 코드까지 섭렵하며 작품에 응용하고 있다.

프로그래밍 창작 기법의 역사는 1960년대 벨 연구소(Bell Labs)로 거슬러 올라간다. 미국의 마이클 놀(A.Michael Noll)과 독일의 프리데 나케(Frieder Nake), 게오르그 네스(Georg Nees) 등의 인물이 바로 디지털 아트의 선구자이다. 이들은 공통적으로 공학에 대한 전문 지식을 갖추고 있었다.

이처럼 현대 아티스트들이 공학 기술을 작품에 응용하고 있는데도 불구하고 대다수의 사람들은 여전히 예술과 과학이 시소의 양 끝에 위치한 것처럼 대척점에 있다고 생각한다. 과학기술은 데이터를 매개로 운용되고 예술은 정서(情緖)를 바탕으로 창작되며, 과학기술은 기술을 통해 발전하는 반면 예술은 개인이 느끼는 주관(主觀)을 표현하며 발전한다고 생각한다. 그러나 오늘날에는 예술과 과학이 긴밀히 결합하여 새로운 형

태의 예술을 만들고 있다. 컴퓨터 분야에 정통하지 않은 일반인이나 예술가들도 컴퓨터 프로그래밍 언어를 사용하여 어렵지 않게 매우 가치 있는 예술품을 만들어낸다.

사실 13세기 초부터 인류는 “프로그래밍이 가능한 규칙(programmable device)”을 연구하기 시작했다. 1차 산업혁명 시기, 프랑스 발명가 조셉 마리 자카드(Joseph Marie Jacquard)는 천공카드를 사용해 직기 프로그래밍 명령을 내리는 방식으로 벽에 거는 융단에 “hello, world” 글자를 직조했다. 1842년에는 프로그래머의 시조 영국인 에이다 러브레이스(Ada Lovelace)가 세계 최초로 프로그램을 작성했다.

2차 세계대전 시기에는 앨런 튜링(Alan Turing)이 프로그램 언어의 최종 형태인 튜링 기계를 발명했다. 그 후 프로그래밍 언어는 포트란(FORTRAN), 리스프(Lisp)에서 초보자들을 위해 설계된 비구조화 형태의 프로그램 언어인 베이직(BASIC) 언어로 발전한다. 1972년에는 데니스 리치(Dennis Ritchie)가 C언어를 발명하고 알랭 콜메르(Alain Colmerauer)는 논리식 프로그래밍 언어 프롤로그(Prolog)를 설계하였다. 1983년에는 비야네 스트롭스트룹(Bjarne Stroustrup)이 C++ 언어를 만들었다. 1995년 제임스 고슬링(James Gosling)이 자바를 발명하고 1996년에는 브렌던 아이크(Brendan Eich)가 자바스크립트 LiveScript를 만들었다.

지금에 이르기까지 많은 프로그램이 개발되어 일반인들의 프로그래밍 언어 학습을 도왔다. 2001년 MIT 미디어랩의 케이시 리이스(Casey Reas)와 벤 프라이(Benjamin Fry)는 Java 언어를 기반으로 프로세싱(Processing) 언어를 구축하였다. 그 외에도 프로세싱을 기반으로 하는 Javascript 프레임 p5.js이 만들어졌으며 2005년에는 재커리 리버만(Zachary Lieberman) 파슨스 디자인 스쿨 교수가 C++를 기반으로 오픈소스 언어 오픈프레임웍스(openframeworks)를 구축했다.

이러한 언어들의 개발로 아티스트들도 간단한 문법과 그래픽 프로그래밍 시스템으로 작품을 창작할 수 있게 되었다. 해당 언어들을 사용하여 진행하는 “Creative Programming (크리에이티브 프로그래밍)”은 모던

디자인과 모던 아트 발전의 이정표가 되었다.

2.3.1. 크리에이티브 프로그래밍 과 예술

컴퓨터 프로그래밍 언어의 본질이 문제를 사고하는 'logic'과 사유방식에 있는 반면 예술 창작의 본질은 문제를 발견하고 이를 바라보는 관점에 대한 묘사 및 개인의 감정을 표현하는 데에 있다. 컴퓨터 프로그래밍 언어는 디자인 도구 외에도 예술을 표현하는 하나의 방식으로써 거듭나고 있다. 예술 디자인과 컴퓨터 기술이 융합하면서 다채로운 예술 표현의 가능성은 대폭 확대되었고, 아티스트는 한층 더 다양한 차원에서 예술을 창작하고 감정을 표현할 수 있게 되었다. 디자인과 프로그래밍의 만남으로 여러 제약에서 자유로워지면서 다양한 형태의 예술 작품이 창작되고 있으며 더불어 과학의 발전도 이루어지고 있다.

컴퓨터 프로그래밍 분야의 언어는 매우 복잡하고 정교하다. 이런 이미지에 대한 영향으로 창의적인 프로그래밍(Creative Programming)을 과학이자 컴퓨터 분야의 산물이라고 여기기 쉽지만 사실 창의적인 코딩(Creative Cording)은 일종의 디자인으로 예술에 더 가깝다고 말할 수 있다. 컴퓨터와 호흡하며 작품을 창작하기 때문에 신체를 도구로 삼아 작품을 진행하던 고전적인 방식은 이제 필요 없어진다. 예술 창작이 두 손에서 자유로워지고 제작 과정과 프로그래밍 과정이 한 공간 안에서 이루어지면서 새로운 창작 예술의 형태가 파생된 것이다. 프로그래밍을 통해 작품 창작 환경이 "현실 공간"에서 "가상 공간"으로 바뀌면서 재미있고 창의적인 방식으로 움직이는(motion) 형태의 예술이 다채롭게 창조된다. 아티스트는 새로운 예술 형태와 창작 수단을 통해 자신을 표현하고, 알고리즘과 프로그래밍을 활용하여 참신하면서도 독특한 예술 작품을 창작한다. 기술이 콘텐츠를 표현하는 방식에 새로운 장을 열어주었다.

2.3.2. 디자이너의 프로그래밍 환경 적응

앞으로는 디자인이나 창작 아이디어 구상 시에도 아티스트가 컴퓨터 프로그래밍을 그 과정에 포함시켜야 한다고 생각한다. 기존의 디자인 과정은 움직이지 않는 정(靜)적인 형태였기에 디자인 전 과정의 통제가 가능하고 디자인 결과도 예상이 가능했다. 이와는 달리 컴퓨터 프로그래밍으로 디자인하는 단계로 들어서면 아티스트의 역할은 전략을 짜는 기획자에 더 가까워진다. 우리가 여러 규칙을 제정하면 구상은 실제로 완성되겠지만, 최종적인 결과에 대해서는 더 이상의 예측은 불가능해질 것이다. 디자인 과정의 다선적(多線) 사유 모델은 비선적(非線)사유 모델에 의해 점차 대체되고 있다. 컴퓨터 프로그래밍 언어로 디자인을 제작하는 과정에서 “창작”은 더 이상 명령을 받고 수행하는 하향식의 프로그래밍 과정이 아니다. 다수의 불확실한 요소들이 디바이스의 참여자에게 놓이면서 디자이너와 아티스트의 절대적인 권위는 사라지고 그를 대신해 창작자와 참여자 사이의 복잡한 관계가 나타난다.

2.3.3. 크리에이티브 프로그래밍의 도구

크리에이티브 프로그래밍에는 “텍스트식 프로그래밍 언어”와 “노드식 프로그래밍 언어” 두 가지 형식이 포함되어 있다. 노드식 언어로는 Pure Data, MaxMSP, vvvv, 노드박스 NODEBOX, Blender, TouchDesigner, Grasshopper 등이 있는데 비주얼 프로그래밍 언어로 모듈을 드래그하여 컴포지션에 프로그램을 만들 수 있다. 텍스트식 프로그래밍 언어에는 프로세싱(Processing), P5, 페이퍼.js, OpenFrameworks, Cinder, OpenRNDR 등이 있다. 오픈소스 프로그래밍 언어로 간단한 언어 연산과 명령 창을 사용한다.

본고에서는 프로세싱(Processing)을 핵심 개발 도구로 삼아 작품 창작을 진행하였다. 프로세싱은 Creative Programming에 매우 광범위하게 응용되는 언어이자 초보자가 입문하기에도 가장 적합한 프로그래밍 언어

이기도 하다. 프로세싱은 디자이너와 아티스트를 위해 만들어진 프로그래밍 언어라고 말할 수 있다. 간단하면서도 사용이 용이한 특성으로 누구나 빠르게 참신한 아이디어를 표현할 수 있다. 게다가 다른 텍스트식 프로그래밍 언어 도구와 비교해봐도 프로세싱은 2001년 출시 후 십여 년이 넘게 쓰이고 있어 사용자의 규모도 상대적으로 방대할 뿐만 아니라 예제 파일도 잘 구성되어 있으며 다양한 라이브러리와 언어 지원도 갖추고 있다. 애니메이션, 비주얼 (visual), 설치예술, 게임 디자인 및 관련 분야에서 적지 않게 응용되고 있으며 웹사이트에는 초보 입문자들이 무료로 사용할 수 있는 리소스도 매우 다양하게 제공하고 있다.

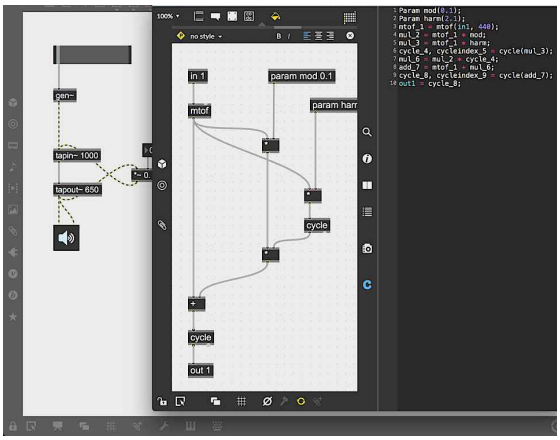


Figure 6. 노드식 프로그래밍 언어 MaxMSP

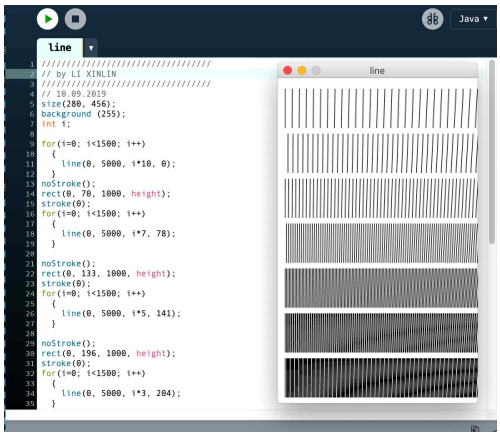


Figure 7. 텍스트식 프로그래밍 언어 프로세싱

프로세싱은 자바 가상 머신(Java Virtual Machine, JVM)를 기반으로 하는 프로그래밍 언어로 GNU/Linux® 및 Mac OS X와 Windows®에 운용이 가능하다. Processing의 IDE는 Processing Development Environment를 줄여 PDE라 부른다. 현재 다수의 언어가 지원되고 있으며 Javascript, Python, renjin 등이 이에 포함된다. 프로세싱은 그래픽스의 sketchbook과 환경을 개발하는 데 목표를 두고 출시되었다. 이미지를 이용해 컴퓨터 과학의 기초 지식을 교육하였는데 시간이 지나며 이미지 시각화 전문 프로그램을 구축하는 환경으로 점차 그 용도가 바뀌었다.

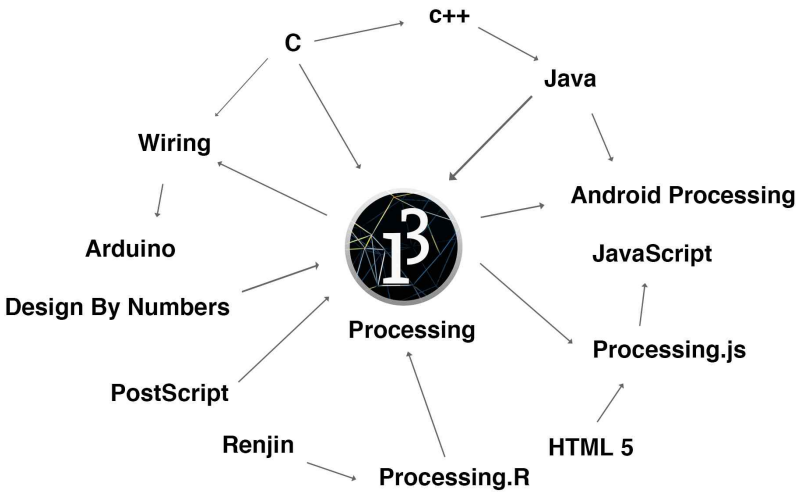


Figure 8. 프로세싱 계통도

2.3.4. 컴퓨터 프로그래밍 기법의 이해

창작자가 컴퓨터 프로그래밍을 사용하여 로직(logic)으로 조합된 자율성(또는 반자율성)을 지닌 시스템을 설계하면, 해당 시스템은 창작자의 의도에 따라 작업을 수행하고 독특한 질감의 결과물을 생성한다. 창작자의 전문 분야, 학습 방법, 사고방식 등의 차이로 크리에이티브 프로그래밍의 방법은 매우 다양해질 수 있다.

본 연구자는 보통 “코드 작성” 전에 아이디어 “계량화”를 진행하거나 구상을 세분화하여 대략적으로 진행할 수 있는 기능 모듈을 만든다. 그런 다음 각각의 모듈을 다시 구체적인 실행 단계 과정으로 세분화한다. 컴퓨터 전문 지식이 부족하더라도 디자인 흐름도(Flow Chart)를 보조 도구로 삼으면 논리적 사고를 할 수 있다. 흐름도를 설계할 때에는 다양한 구조 유형, 모듈 간 작업 흐름 방향, 인풋 아웃풋 등을 반영할 수 있다. “아이디어 → 단계 설계 → 논리 결함 발견 → 수정”의 과정을 통

해 논리 사고를 반복하여 검토한다.

그 밖에도 작품 창작 이전에 알고리즘이나 기하학 모형을 택하여 확산적(divergent)인 R&D 작업을 하며 그것이 갖고 있는 특성들이 시각적으로 표현될 수 있는 여러 가능성을 탐색한다. 이러한 R&D 작업의 성과는 최종적으로 창작의 "창고"가 되어 적합한 주제를 만났을 때 선택적으로 사용할 수 있다.

프로그램으로 작품 창작할 때에는 알고리즘 말고도 데이터, 물리 등 다른 분야의 지식에 관한 학습이 필요하다. 비전공 분야의 지식을 섭렵할 때는 각종 장애물과 한계에 부딪칠 수밖에 없다. 많은 시간과 노력을 할애하여 스스로 어려움을 극복하고 경험과 노하우를 쌓는 작업은 크리에이티브 프로그래밍에서 필요한 과정이다.

3. 생동하는 타이포그래피

3.1 키네틱 타이포그래피의 역사

3.2 키네틱 타이포그래피의 구조

3.3 키네틱 타이포그래피의 디자인 특징

3.4 키네틱 한자의 디자인 특징

제3장 생동하는 타이포그래피

3.1 키네틱 타이포그래피의 역사

문자의 발전사를 살펴보면 문자는 모두 손글씨(handwriting)에서 활자(type)로 변화 발전했다는 사실을 알 수 있다. 영문 중 타이포그래피(Typograph)와 관련된 “활자 디자인(type design)”과“활자 꼴 디자인(typeface design)”에서도 유사한 양상을 보인다. 이를 통해 타입과 인쇄의 관계가 매우 긴밀하다는 것을 엿볼 수 있다.

문자는 조판 인쇄, 활자 인쇄, 기계 인쇄 등 일련의 역사적 변화를 거쳤다. 문자는 인쇄되어 서적으로 만들어졌고 또 그것을 묶어 휴대 가능한 형태로 여러 사람이 나눌 수 있게 되면서 그 과정에서 정보, 관념, 지식과 같이 과거 일부 소수만 독점했던 정신적 가치재가 인쇄를 통해 다수의 대중에게 전해질 수 있게 되었다. 인쇄 문자는 이제 그 자체로서 가치를 지닌 ‘신성한’매개가 되었다. 구텐베르크(Gutenberg)의 활자 사용을 시작으로 인쇄는 제작과 디자인과 밀접하게 결합하여 정보를 전달하는 하나의 방식이 된다. 인쇄기술이 하나의 새로운 예술이 되었다고 말할 수 있는데 이것이 바로 타이포그래피(typography)이다. [15][16]

오늘날의 인쇄 텍스트(text)는 전반적으로 여전히 17세기 이전의 원칙과 질서를 그대로 이어가고 있다. 인간의 창작으로 만들어진 서면 문자는 보통 책 표지, 제목, 목차, 장과 절, 색인 등 낱장으로 인쇄되어 책으

---

15) 타이포그래피(typography)는 폰트, 문자 크기, 행간, 자간, 들여쓰기 등 조판 편집 배열과 관련된 기술부터 종이 재질, 책의 겉모양을 꾸미는 장정까지 전 과정을 아우르는 예술 형태를 말한다. 서적 조판, 포스터 디자인, 광고, 캘리그패피(Calligraphy), 그라피티(graffiti), 탁본, 공공 장소 표시, 상표 등 문자의 사용이 필요한 공간은 모두 타이포그래피와 밀접한 관계 보인다. -『Type face 산책《字型散步》』.p20-21.

16) “타이포그라피란 결국 텍스트의 의미와 시각이미지의 관계를 다루는 행위이다. 타이포그라피는 글자의 모양을 선택하고, 그것들을 배열한다.”- 박한수, <타이포그라피 공간감을 통한 텍스트의 구체화 연구>. p5.

로 묶인다. 문서는 일단 인쇄되어 나오면 그 구조와 외부 형식이 고정되는데 시간에 따라 변하는 것은 읽는 순서로 독서 시간은 오로지 독자에 의해 결정된다.

한편 해당 기술이 문화와 만나는 과정에서 시간 기반 미디어(TIME-BASED MEDIA)는 타이포그래피에 재미 요소를 더하며 변화를 가져왔다. 우리는 주변의 영화, TV, 컴퓨터 영상 등 각종 영상 자료를 통해 새로운 방식의 녹음 제작, 편집, 전송 및 정보 수용 시스템이 천여 년 동안 동일한 형태를 유지하고 있는 "문자"에 영향을 미치고 있는 사실을 엿볼 수 있다.

움직이는 문자의 역사는 영화 타이틀 시퀀스(film title sequences)와도 관련되어 있다. 초기 영화의 오프닝 제목, 영화 TV 프로그램의 배우와 스태프 명단, 광고 상품이나 상표 내용의 묘사 등도 모두 시간이 흐르면서 텍스트를 이용해 정보를 전달하는 방식을 사용하고 있다. 다수의 광고 회사들이 상품의 브랜드 가치를 잘 전달하고 대중의 관심을 불러일으키기 위해 적극적으로 스크린 안에 타이포그래피(on-screen typography)를 사용하면서 발전이 추진되었다. 시간 기반 미디어 속에서 타이포그래피는 물리적인 인쇄 공간에서 탈피한다. 글꼴, 두께, 크기, 색상 등과 같은 정적인 속성은 "움직임 동작(행위)"에 자리를 내어준다. 깜박임, 오리엔티어링, 휘거나 변하는 등 시간과 관련되는 동작은 모두 평면 디자이너들이 별도로 사용하는 표현 도구가 되었다. 스크린 위에서 시간의 영향을 직접적으로 받는 조판 편집을 우리는 "시간적인 타이포그래피(Temporal typography)"이라고 통칭한다.

시간 타이포그래피 중에서 동역학(kineticism)을 사용하여 표현하는 기법은 매우 흔하게 볼 수 있다. 점프하고 춤을 추는 자모음은 즐거움을 전달하고, 힘없이 느린 신호로 슬픔을 전달하고, 덜덜 떨리듯 흔들리는 신호로 공포를 표현할 수 있다. 단어 자체에 포함하고 있는 언어적 의미 외에, 이러한 정보들 또한 표현될 수 있다.

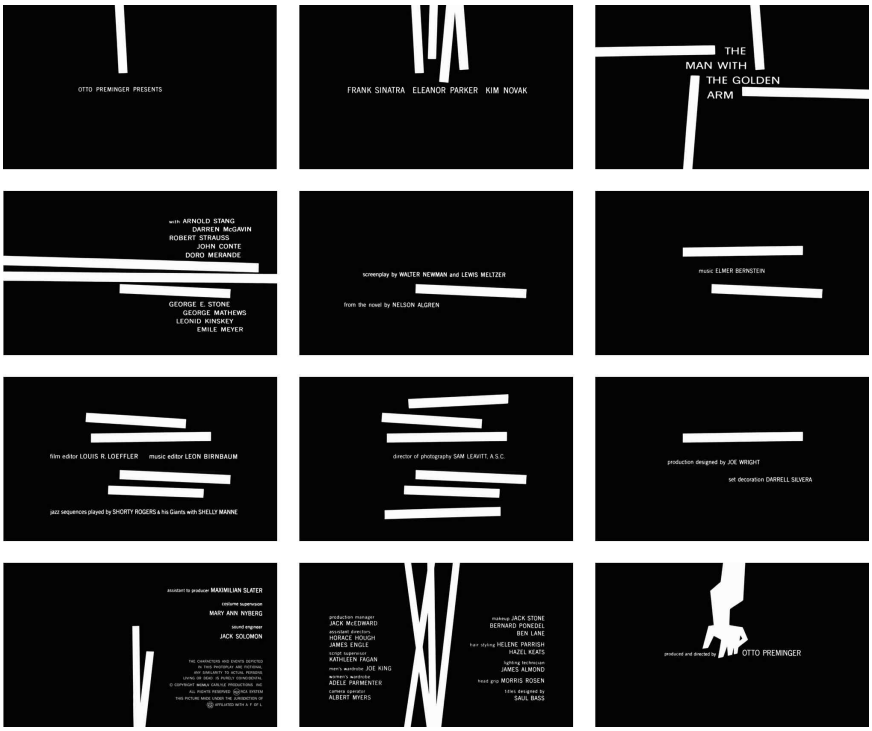


Figure 9. Saul bass, Title sequence, <The Man with the Golden Arm title sequence>,1995

이러한 표현기법은 가장 먼저 스톱 모션 애니메이션(stop-motion animation)에서 나왔는데 1899년 조르주 멜리에스가 창작한 광고에서 최초로 움직이는 자모의 형태로 등장하였다. 1903년 에드윈 포터(Edwin S. Porter's)가 감독한 영화 <톰 캐빈Tom's Cabin>에서는 처음으로 자막 카드(intertitles) 의 개념이 도입됐다. 1950년대 이후에는 고정 형식을 따르던 타이틀 자막(title sequences)이 새롭게 바뀐다. 솔 바스(Saul Bass)와 오토 프레밍거(Otto Preminger)가 1954년 창작한 <카르멘 존스 Carmen Jones> 오프닝 크레딧을 시작으로 정지 상태의 텍스트 리스트도 움직이기 시작한다.[17] 후에 노먼 맥라렌(Norman McLaren), 모리스 바인더(Maurice Binder) 등 인물도 키네틱 타이포그래피 분야에 지속적

17) Mark Sanders,Sandra Maxa,Ben Day,Philip B.Meggs,Rob Carter, John Wiley&Sons <Typographic Design: Form and Communication> 6th Edition. 2014, Cap.9, Vs 2.

인 탐구를 진행한다. 이들은 모두 TV 광고, 단편 및 영화 타이틀 시퀀스 등 분야의 디자인에 종사하였는데 애니메이션 기술과 그래픽스 디자인 법칙을 서로 결합하여 “수작업” 형태의 방식으로 다양한 폰트 만화를 영상과 결합하였다.

그러나 앞서 예로 제시했던 초기 작품들은 디지털 방식으로 만들어진 것이 아니라는 공통점이 있다. 현대의 시간 편집(Temporal typography) 타이포그래피는 영화와 TV 프로그램뿐만 아니라 광범위한 디지털 미디어도 그 대상에 포함한다. 1960년대를 시작으로 아티스트들은 이제 “움직이는 텍스트(moving text)”를 각종 전자 기기와 디지털 기술에 혼합하기 시작한다. 대표적인 예로 스탠 밴더빅(Stan Vanderbeek) 과켄 놀턴(Ken Knowlton)이 컴퓨터로 만든 타이포그래픽 애니메이션 <Poem Fields> [18] 가 있다. 그 외에도 그린버그 어소시에이트(R/Greenberg Associates) 회사가 참여 제작한 영화 타이틀 시퀀스 및 특수 효과가 있다. 그리고 인터랙티브 분야의 유명 디지털 미디어 아티스트 존 마에(John Maeda)가 이끄는 MIT Media Laboratory 등등이 있다. 이처럼 움직이는 형태의 폰트를 디지털 미디어 환경에 노출시키면서 걸출한 디자이너와 디자인팀이 연이어 쏟아져 나왔다.

3.2 키네틱 타이포그래피의 조직구조

키네틱 타이포그래피의 역사적 과정을 살펴보는 것과 동시에 우리는 그에 대한 정확한 이론적인 분석도 할 필요가 있다. 해당 용어들의 특징을 나누어 구체적으로 살펴보고 전체적인 맥락에 대해서 정리를 해보겠다. 현재 “Temporal typography” 분야는 하나의 완전한 시스템을 갖추고 있으며 “움직임”이나 “동역학(kineticism)” 묘사에 사용되고 있다. 이에 대해 바바라 브라우니(Barbara Brownie) 는 『Transforming Type: New Directions in Kinetic Typography』에서 유형별 동역학에 대해 구

18) <Poem Fields> (1964 - 6), computer-generated typographic animations.

분지었는데 아래에 “Temporal typography”를 정리한 그림 10) 이 제시되어 있다.

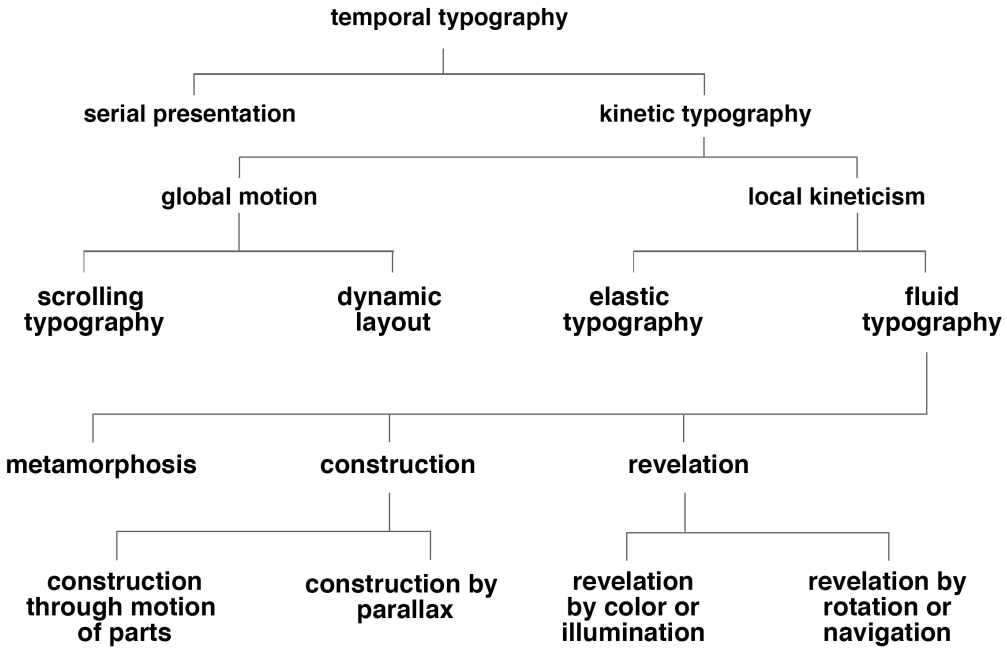


Figure 10. Temporal typography의 분류

연속 프레젠테이션 (serial presentation) 은 시간의 “연속성”을 사용하여 콘텐츠에 연속적으로 나타나게 하는 것으로 텍스트 자체는 정지되어 움직이지 않는다. 인지심리학 연구 중의 RSVP(rapid serial visual presentation, 신속 순차 시각 제시)로 구체적인 표현이 가능하고, 테스트 중인 문자는 “동일한 위치에서 연속적으로 표시”되며 이동하거나 바뀌지 않는다. “연속 프레젠테이션(serial presentation)”도 “시간 타이포그래피(Temporal typography)”에 포함되지만 본고에서는 주로 “이동” 또는 “변화”하는 키네틱 타이포그래피에 대해 논하겠다.

스크롤링 타이포그래피와 다이내믹 레이아웃(scrolling and dynamic latout)의 대상은 한 위치에서 다른 위치로 이동이 가능하고 회전 각도도 바꿀 수 있다. 개별 폰트의 디자인과 동작보다는 주로 콘텐츠의 분포와

구조에 포커스를 둔다. 여기에서 동역학은 오리엔터링(위치 이동)에 국한되며 자모음 사이의 관계만을 고려할 뿐 개별 독립 폰트의 특징이나 본질은 아니다. 다수의 학술 문서에서 키네틱 타이포그래피는 “이동”하거나 “시간이 변화”하는 “텍스트”를 사용하는 것으로 정의된다. [19] 초기 키네틱 타이포그래피 작품은 대부분 이러한 유형에 집중되었다. 그림 11) 과 같이 로마자 알파벳 또는 단어가 스크린 상에서 위치는 바뀌나 개별 자모음의 속성은 바뀌지 않는 것을 예로 들 수 있겠다. 개별 단어의 이동이나 새로 배열 조합된 것이든 아니면 다른 차원(공간)에 들어선 것이든 개별 폰트는 형태, 색상, 자모음의 의미와 같은 맨 처음의 고유 특성을 계속해서 가진다.

Figure 11. Jacob Gi;breath, Oklahoma State University, 2011, O’Brien Kinetic Typography.

본문에서는 “타이포그래피(typography)” 용어를 글자꼴(letterforms)의 배열과 외관을 묘사하는데 사용하였다. 이에 따라 키네틱 타이포그래피의 논의 도 “배열”에만 국한하지 않는다. 알파벳은 단어나 구절로 된 문

19) “Kinetic typography - text that uses movement or other temporal change”- Johnny C. Lee, Jodi Forlizzi, Scott E. Hudson <The Kinetic Typography Engine: An Extensible System for Animating Expressive Text>

장 안에 있을 때에만 가치를 지니는 것은 아니다. 개별 알파벳 또한 독립된 생명을 가진다. 그림 10) 에 제시된 키네틱 타이포그래피의 갈래 “유동 타이포그래피 (fluid typography)” 처럼 알파벳도 탄성, 휨, 중첩, 은유 등 다양한 형태로 표현되며 가치를 지니고 이를 통해 전체 작품의 성격에도 영향을 준다.

3.3 키네틱 타이포그래피의 디자인 특징

문자의 중첩, 선명도, 각도의 변화, 플립 북(flip book) 등의 방법을 통해 종이 매체와 같은 인쇄물에 움직이는 효과를 표현해낼 수 있지만 키네틱 타이포그래피는 디자이너에게 콘텐츠와 시각적 형식 및 행동을 연결하는 직접적인 작업의 기회를 제공해준다. 이에 편집 배열의 기본 요소를 고려하는 것 외에도 디자이너들은 스크린 속 문자의 움직임과 동작도 결정해야 한다. 현재 키네틱 타이포그래피와 일반 타이포그래피를 나누는 디자인의 기준은 매우 다양하다. 시간 특성으로 구분하거나, 움직이는 스타일로 기준을 나누거나 또는 문자 자체가 내포하는 의미를 기준으로 나누기도 한다. 위의 종합적인 분석을 토대로 아래에 디자인 특성 세 가지를 도출해보았다.

3.3.1. 다차원적 특성

폴 클레(Paul Klee)와 칸딘스키(Kandinsky)는 디자인의 개념요소에 대해 “선은 점으로부터 출발한다”고 정의하며, “모든 형태는 움직이는 점에서 시작된다”고 말했다. 즉 점이 움직여 선이 되고, 선이 움직여 면이 되며, 면이 움직여 입체가 된다. [20]

20) 원유홍, <타이포그래피 천일야화>, p48.

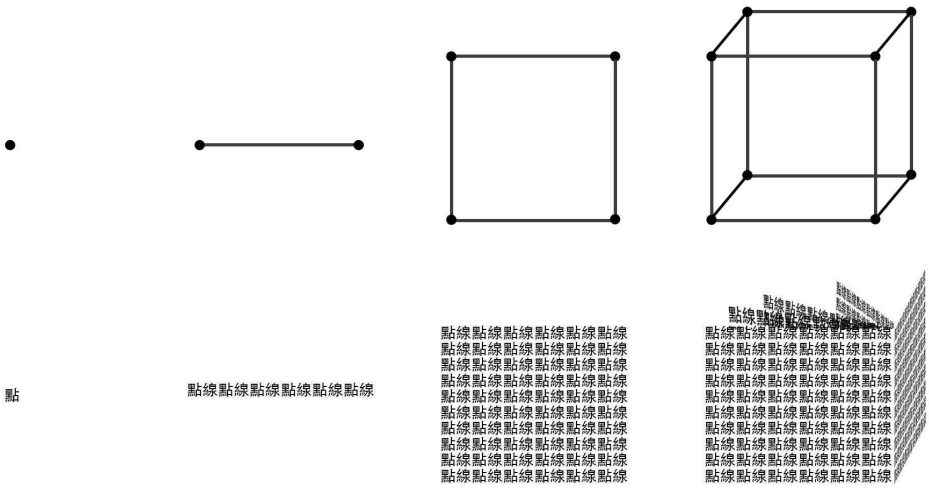


Figure 12. 타이포그래피의 다차원적 특성

문자는 2차원 세계의 산물이다. 디자이너는 이 평면적 제약을 뛰어넘기 위해 많은 노력을 시도해왔다. 분할, 중첩, 재구성 등의 방법을 시도하여 만든 거리감과 공간감이 바로 그 예이다. 그러나 비록 2차원에서 공간의 개념을 표현해냈다 하더라도 2, 3차원에서 4차원으로 전환되는 과정에서 평면 문자의 '시간적 특성'제약에서 벗어나기란 쉽지 않다.

마찬가지로 3차원 공간이나 디지털 미디어에서 우리는 글자의 변화나 외부환경의 변화를 통해 더 용이하게 움직임의 시각적 형태와 느낌을 자아내지만, 해당 움직임의 개념도 사실 시간의 개념에서 생성된 것이다. 시간 특성은 2차원 평면 디자인에서는 암시될 뿐이지만 디지털 미디어 환경에서는 움직이는 방식을 통해 '시간적 특성'을 충분히 표현해낼 수 있고 실제로도 대중에게 재현되고 있다. 여기서 말하는 시간의 특성은 주로 작품 안의 시간축 개념, 컨텐츠의 패턴 등으로 표현되는 것을 말한다. 이에 본 연구자는 시간의 추이에 따라 형성된 글자의 변화가 바로 키네틱 타이포그래피이고 이것이 평면 문자 디자인의 속성과 구별되는 특징 중 하나라고 생각한다

3.3.2. 운동 특성

운동 시각은 감각 물체의 공간 이동과 이동 속도의 변화를 인지하는 기능을 의미한다. 움직임은 인간의 시각적 주의 집중을 가장 쉽게 유발하는 현상으로 지속성을 가지고 움직이는 운동 과정이 있으면 관람자는 이를 더 쉽게 기억한다. 이것이 바로 대다수의 사람이 정지 상태의 서적보다 움직이는 TV 영상을 더 흥미롭게 바라보는 이유이다. 애니메이션과 영화 등 시각 미디어를 구성하는 바탕은 바로 시각 머무름 (Visual staying phenomenon, duration of vision) 현상으로 이것은 인간이 물체를 감지하는 기본적 생리현상이다.

정지 상태의 평면 시각으로 표현한 타이포그래피 디자인을 '사물'이라 상정한다면 움직이는 상태로 표현한 폰트는 '사건'이라 할 수 있다. 우리에게는 '사물'보다 '사건'이 더 쉽게 본능적으로 각인된다.

움직이는 글자를 디자인할 때에도 우리는 자연 물체 운동의 일반적인 규칙을 따라야 한다. 물체 낙하 운동, 관성 운동, 원상태로 돌아가는 운동 등이 그 예로 자연 물체는 모두 고유의 특수 성질을 가진 운동에 영향을 받는다. 글자에 움직임을 가할 때에도 이러한 점을 반영해야 보다 실재적으로 느껴질 수 있다. 이러한 운동 특징이 바로 키네틱 폰트와 평면 폰트 간의 본질적인 차이를 만드는 요소이다.

3.3.3. 기술의 다원화 특성

편집 디자인 문자에서 재질을 활용하는 것은 간과해서는 안 되는 요소이다. 하지만 다른 재질로 표현하는 효과가 비록 다양성의 특징을 가지고 있다 하더라도 재질을 활용하여 표현하는 것은 모두 '움직임이 없는 정지 그래픽'의 공통성을 갖는다. 이러한 그래픽은 소재 선택을 달리한다고 해서 시간상으로나 동태적으로 변화하지 않는다. 왜냐하면 움직이는 글자는 '시간'과 '운동'의 특징을 가지고 있고 또 최신 기술을 활용하여 더 큰 효과를 이끌어낼 수 있기 때문이다. 최신 기술의 예로 컴퓨터 소

프트웨어 기술, 시각화 네트워크 (Visualization), 동영상, 교차 호환 디바이스 등을 들 수 있다. 이러한 신기술 수단으로 움직이는 글자 디자인은 각종 미디어에서 다양하게 표현될 수 있다. 작품의 인터랙티브(양방향) 특성을 더해주는 것과 단순한 시각 감지에서 시각, 청각, 후각, 촉각, 미각의 오감을 감지하는 특성으로 전환시키는 것이 바로 그 예이다.

3.4 키네틱 한자의 디자인 특징

3.1 부분에서 우리는 키네틱 타이포그래피 디자인이 기존의 타이포그래피 디자인과 구별되는 몇 가지 특징에 대해 살펴보았다. 이러한 특성은 한자를 비롯한 다양한 문자의 키네틱 타이포그래피 디자인에도 활용이 가능하다. 각각의 문자에 키네틱 타이포그래피 디자인을 하는 것은 보편적인 가지와 의의를 지닌다. 다만 우리가 주지해야 할 사항은 한자의 형태와 창조 방법이 여타의 문자와 크게 다르다는 점이다. 이에 본 연구자는 한자의 특수성을 구체적으로 분석하는 작업이 한자 키네틱 디자인의 이론적 토대를 형성하는 데에 도움이 될 것이라고 생각되어 3.2 장에서 보다 구체적으로 한자의 특징을 살펴보고자 한다.

3.4.1 독특한 한자 체계

세계 고대 문자는 모두 초기에 그림으로 내용을 기록하는 “그림문자(pictograph)”의 형태로 출발한다. 그다음으로 그림에 언어가 결합하면서 사물의 형태를 간략하게 그림으로 표현한 상형 기호로 단어와 문장을 기록하게 되는데 상형 문자는 후에 여러 문자 체계로 발전한다.

Figure 13. 창위안 암벽화, 짐승의 형상으로 표현한 의식도

문자의 기호는 표형법(表形, pictogram)과 표음법(表音, ideogram) 그리고 표형과 표음을 합한 세 가지 방법으로 나눌 수 있다. 문자 발전의 단계에 대한 설명은 아래의 표에 제시되어있다.[21]

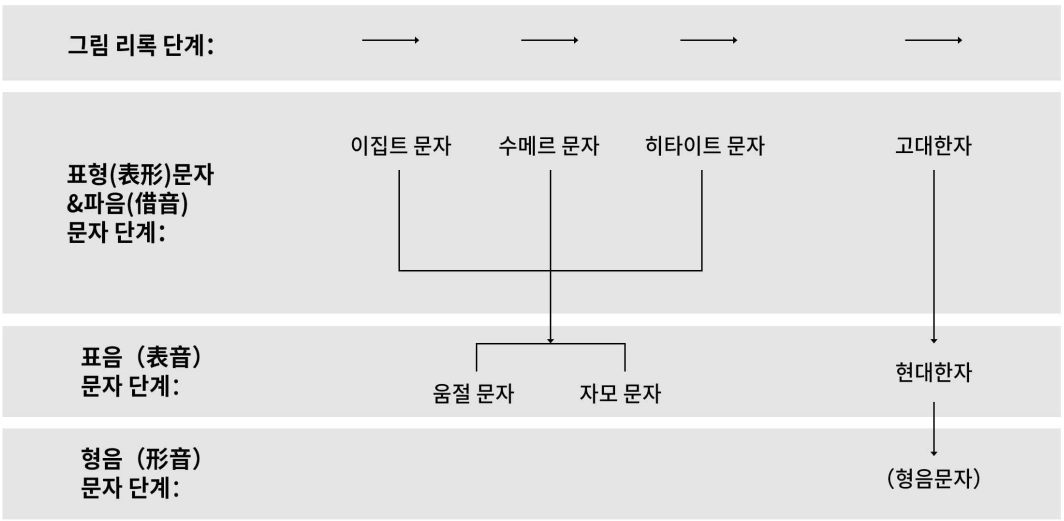


Figure 14. 문자 발전 단계표

고대 이집트 “히에로글리프 문자”, 메소포타미아 “수메르 설형문자”,

21) 劉又辛, 方有國 <An Outlinr of the History of the Development of the Chinese Characters>, 1999, Cap.2, Vs 3.

중국 상대 “갑골문자”는 모두 그림으로 내용을 기록하는 원시 형태에서 발전한 고대 문자로 석기 시대에 기록한 그림과는 본질적으로 달랐지만 아직 완전한 문자는 아니었다. 해당 고대 문자들은 계속 발전을 거듭하여 이집트 고대 문자와 메소포타미아 설형문자는 자모 형태의 표음문자로 발전한다. 현재 세계 대부분의 문자는 모두 몇 십 개의 자모음으로 글자와 소리를 기록하는 표음문자의 형태를 보인다. 표음 문자는 “形(형상)” 대신 음표(音符)로 글자를 기록한다.

이와는 달리 한자는 표의문자로 발전한다. “形(형상)”는 그림 문자가 발전하여 만들어진 것인데 이 형상에 발음을 표시하는 음이 더해져 “형상문자”가 만들어진다. 형상문자는 “표형(表形) +표음(表音) ”이 결합한 문자로 한자 체계는 이로부터 발전하여 완성되었다.

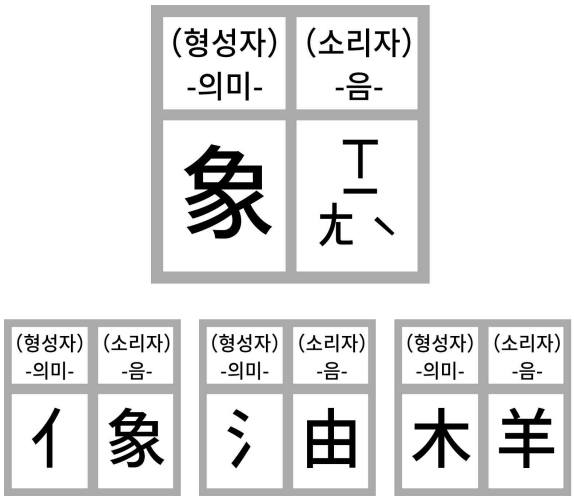


Figure 15. 모양 상 “像”, 기름 유“油”, 모양 양 “樣”/ 상수리나무 상“像”

한자 서체는 “대전(大篆)”에서 “소전(小篆)”으로, “소전(小篆)”에서 “예서(隸書)”로, “예서(隸書)”에서 또다시 “해서(楷書)”로 발전한다. “대전”에서 “소전”으로 바뀌기까지는 천여 년의 시간이 걸렸는데 그 과정에서 변화를 거치며 차츰차츰 완성되었다. “대전(大篆)”의 글자 모양은 사물의 형상에 가까운데 다수의 글자가 아직까지도 색채를 유지하고 있다. “소

전(小篆)”에서는 둥글둥글하고 가지런한 모습으로 바뀌고 사물의 원형에서 벗어난 추상적 형태를 보인다. “소전(小篆)”에서 “예서(隸書)”로 바뀌는 과정에서 한자 서체는 크게 달라진다. “소전”의 둥글둥글한 원형의 윤곽은 모두 정방형으로 바뀌고 곡선의 선도 각기 가로 (一) , 세로 (|) , 점 (丶) , 왼쪽 삐침 (ノ), 오른쪽 삐침(乀) 등 필획으로 변모한다. “예서(隸書)” 이후로 한자는 완전히 네모난 글자를 뜻하는 “方塊字”의 정방형 문자로 바뀐다. 현대에 사용하는 한자 역시 “方塊字”의 기본 구조를 그대로 따르고 있다.

한자 체계는 총 3 단계를 거쳐 완성되었다. 1단계는 “그림 문자” 단계로 상대(商代) 이전의 문자가 여기에 속한다. 2단계는 표형(表形)문자를 기반으로 하여 표음(表音)문자가 주를 이룬 “표음문자” 단계로 갑골 문자부터 진나라 시대의 문자 “대전(大篆)”이 모두 이 단계에 속한다. 3단계는 “형성 문자”가 주를 이룬 단계로 표형문자와 표음문자가 결합한 “형음(形音)문자”도 함께 포함된다. 진한(秦漢) 시기의 “소전(小篆)”, “예서(隸書)”, “해서(楷書)”부터 현대 한자가 모두 이 3단계에 속한다.

Figure 16. 한자의 발전 단계표

한자는 또한 독특한 창작 방식으로 구성된다. 후한 학자 허신(許愼) 은 《설문해자》에서 한자의 구성 원리를 “상형(象形), 지사(指事), 형성(形聲), 회의(會意), 전주(轉注), 가차(假借)” 여섯 가지로 나누어 설명하였는

데 우리는 이를 통칭해 “육서법(六書法)” 이라 부른다. 후에 다른 학자들도 “사체이용(四体二用)”, “삼서(三書)” 및 “신삼서(新三書)”등[22]의 구성원리를 제시하였다. 그러나 학술 이론이 아무리 다양하게 발전해도 한자의 고유성은 계속 유지되고 있는데, 글자 안에 담긴 의미와 이를 나타내는 “형태”가 긴밀하게 연결되어 있는 점이다. 이러한 한자의 조합 원리를 이해하면 키네틱 한자 디자인의 특징을 알고 새로운 키네틱 한자 작품을 창작하는 데에 큰 도움이 된다.

3.4.2 키네틱 디자인의 측면에서 살펴본 한자의 미

키네틱 한자 디자인 시 우리는 문자 활용 디자인을 고려하는 동시에 한자의 디자인과 결합하는 방안도 살펴보아야 한다. 왜냐하면 한자의 구조적인 특징이 다른 문자와 달리 매우 복잡하기 때문이다. 이는 멀티미디어 환경에서 한자 키네틱 디자인의 발전이 영문 키네틱 디자인 보다 더딘 원인이기도 하다. 한자의 “기호학적 기능”과 “그래픽적 기능” 등 디자인 측면에서 바라본 한자 키네틱 특징에 관한 분석도 매우 부족한 편이다. 이에 본 연구자는 아래에서 이를 분석하여 정리하고자 한다.

3.4.2.1 구조적 특징

a. 한자의 다차원적 특징

키네틱 타이포그래피의 특성과 마찬가지로 움직이는 한자도 다차원적이 특징을 갖추고 있어야 한다. ‘한자’의 다차원적 개념을 제시하는 것은 새로운 각도에서 접근하는 것이다. 한자의 구조적인 특징으로 한자 안에는 다차원적 공간이 내재되어 있는데, 다차원적인 측면에서 한자의 일반

22) 삼서(三書): 1.상형문자, 2 표의문자, 3.형성문자. 신삼서(新三書):1.표형문자, 2.표음문자, 3.표형 겸 표음. - 劉又辛, 方有國 <An Outlinr of the History of the Development of the Chinese Characters>, p79.

적인 형태를 변형시켜 키네틱 한자 디자인을 모색하는 연구는 새로운 시각적인 즐거움을 줄 뿐만 아니라 시각언어인 한자를 보다 풍부하게 만들 수 있어 중요한 의의를 갖는다.

한자는 필획으로 구성되고, 다시 필획이 서로 조합되어 이루어진 '부건(部件)"이 합쳐져 하나의 문자가 완성된다. 필획은 한자의 가장 기본적인 구성 요소로서 결합되는 과정에서 다양한 공간적 차원의 변화를 가져온다. 다차원의 측면에서 독립된 글자 하나를 살펴보면 "필획"은 1차원 세계의 "점"이고, "편방"과 "부수"는 "선"이 된다. 이러한 부분들이 조합되면 최종적으로 글자 하나가 완성되는데 이 "완성된 글자"가 바로 2차원의 "면"이다 [23]. 한자는 "문자(文字)"라고도 부르는데, "문장(文)"과 "글자(字)"의 의미는 서로 다르므로 한자는 "글자"와 "글" 두 부분으로 구성된다고 말할 수 있다. 이에 글 역시 글자(점), 행(선), 문단(면), 칼럼(몸체)을 통해 독특한 "공간" 원리를 표현할 수 있다. 이러한 한자 요소의 결합으로 만들어지는 4차원 공간은 키네틱 한자의 필수적인 디자인적 특징이 된다. 이에 아래에서 1~4차원으로 세분화하여 이를 구체적으로 설명해보겠다.

- 1차원 시점

다차원 측면에서 키네틱 타이포그래피의 디자인 요소를 살펴보면 "필획"이나 "한 개의 글자"는 키네틱 한자를 구성하는 단위세포로 "점"이 바로 이에 해당한다. 이러한 세포들이 모여 "편방 부수"와 "문장"을 조합하는데 이것이 바로 "선"이다.

첫째 "필획"[24]은 일반적으로 한자의 형태를 구성하는 점과 선을 말한

---

23) 한글을 예로 들어 설명해보겠다. 한글은 표음문자인 소리글에 속한다. "낱자"는 한글을 구성하는 최소 단위인데 한자로 따지자면 "편방"과 부수"가 이에 해당한다. 이 "편방"과 "부수"가 모여 "필획"을 구성하므로 "필획"이 바로 한자를 구성하는 최소 단위가 된다.

24) 한자를 구성하는 가장 작은 단위는 필획이다. 붓을 한번 대었다가 뗄 때까지 만들어지는 점이나 선을 한 획이라 한다. "가운데 중(中)"자는 총 4획으로 이 '4획'은 필획의 개수를 말하는 것이다. 필획의 기본 형식은 점과 선이다.

다. 가로획(一) , 세로획(丨) , 점(丶) , 왼쪽 삐침(丿), 꺾임(𠂉)등을 예로 들 수 있겠다. 점과 선은 한자의 글자 모양을 구성하는 최소 단위이다. “필획”이 전체 화면의 한 “점”이라면, 여러 점들을 연결한 “선”은 바로 한자의 “편방 부수”를 말한다. “편방”과 “부수”를 구분하는 건 여러 가지로 나뉘어 본고에서는 이를 구체적으로 다루지는 않겠다. 다만, 시각 조형의 측면에서 살펴볼 수 있는 “辶”, “礻”, “灬”, “厂”, “人”등 구조는 한자 조합에서 리듬감 있는 선을 형성하기에 예로 제시한다. 특히 이러한 리듬감은 행서(行書), 초서(草書)와 같은 변화가 심한 손글씨 서체에서 두드러진다.

둘째 한자를 분할할 수 없는 하나의 덩어리로 전제하고 살펴보면 한 글자나 한 단어는 한 점으로 바라볼 수 있고 “구(句)”는 “선”으로 바라볼 수 있다.

- 2차원 시점

한자는 네모난 글자 “方塊字”로 불리기도 한다. 아래에 제시되어 있는 그림과 같이 모든 한자는 네모난 꼴 안에 일정한 면적을 가지고 있다.

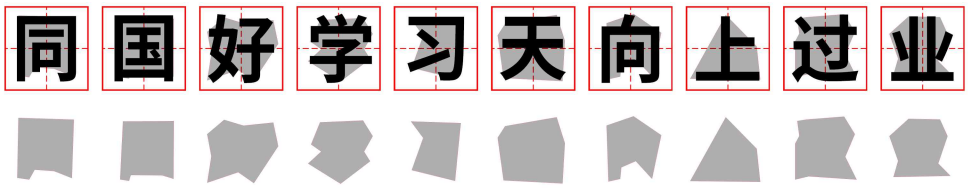


Figure 17. 예: 같을 동 “同”, 나라 국 “國”

문자가 조합된 문장이 여러 개 모이면 “단락”을 구성하는데 페이지 상의 “단락”이나 “문단”은 “면”으로 지각된다. 가지런히 정렬된 단락은 사각형으로 지각되고, 무질서하게 정렬된 단락은 다각형으로 지각된다.

Figure 18. 단락

- 3차원 시점

본문에서 지칭하는 3차원은 2차원에서 가상으로 만든3차원 사물도 포함하는 것을 뜻한다. 예를 들면 조명, 점층, 중첩, 변형, 회전, 가상과 실재, 덩어리 면, 3D 조형 등이 모두 자주 사용하는 제작 기법이다. 이러한 형식 위에서 진행하는 연구는 한자 디자인, 표지 디자인, 포스터 디자인, 편집 배열 디자인, 이미지 디자인에 큰 도움이 된다.

- 4차원 시점

많은 사람들은 4차원 공간을 3차원 공간에서 시간 축을 더한 개념으로 생각한다. 이것이 바로 헤르만 민코프스키(Hermann Minkowski)가 제시한 4차원 시공 개념이다. 평면 안의 타입을 이용해 중량, 공간, 방향, 위치 등 요소로 움직이는 시각 효과를 만들어 낼 수 있지만 본고에서 논의하고자 하는 4차원은 3차원 공간에서 도입한 시간, 운동의 공간 등 개념을 지칭하는 것으로 최종적으로는 시각으로 표현해내는 영상 공간을 말한다.

아래 도표에 위에서 상술한 내용을 간략하게 정리해보았다.

Figure 19. 한자 구조의 다차원 시각 분석

b. 복합성 시점

한자의 구조는 조합되는 유형 방식으로 나눌 수 있다.

일반적으로 한자는 병렬, 한 글자가 다른 글자를 감싸 안는 포위, 글자를 이어 합하는 교합 이 3가지 방식으로 조합된다. 그 밖에도 특수하게 조합되는 방식인 '중첩'과 '합착' 이 있는데 '更' 다시 갱 자와 '重' 무거울 중 자가 바로 그 예이다. 한자는 적게는 2개의 부수로, 많게는 9개의 부수가 결합되어 글자를 형성한다.

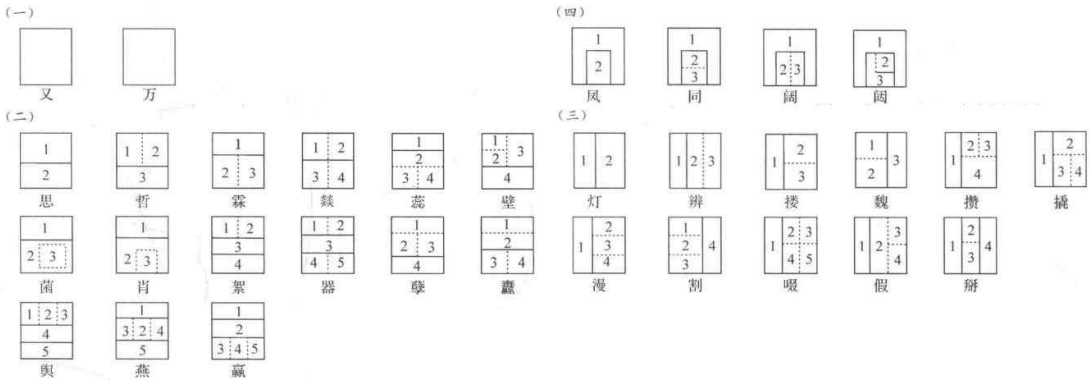


Figure 20. 한자의 구조 종류

c. 모듈화 시점

독일 학자 로타 레더로제(Lothar Ledderos)는 그의 저서 『만물』[25]에서 중국 예술의 공통적 특징인 '모듈'화를 독창적으로 제시하였다. 그는 '조합' 특징을 들어 중국 예술 제작과 서양 예술 제작의 차이점을 구분 지었다.

이 '모듈' 이론은 한자의 글자 조합 방법을 설명해준다. 즉 한자의 필획은 '모듈'을 구성하는 기본 '원소'로 편방 부수와 단일자(單字)로 확장되는 조합 방식을 가진다. (표 1 참조) 이에 모든 한자는 조합되거나 분리되어 단일 글자를 형성할 수 있다. 이와 같은 모듈화 구조 이론은 한자 폰트 디자인의 키네틱 구성 방식과 시각적 표현의 토대가 된다.

원소(element)	단독 필획
모듈(module)	부재 (部材)혹은 구성요소
단위(unit)	단일 한자
계통(series)	연관 한자
총합(mass)	모든 한자

<표 1> 한자의 "모듈"화 시스템

d. 형태의 시각적 시점

시각 디자인의 측면에서 한자의 형태 변화를 분석하면 우리는 어렵지 않게 한자와 서양 문자 체계의 차이점에 대해 알 수 있다. 이러한 차이는 중국식 사유 방식과 가치관을 엿보게 한다.

---

25) Lothar Ledderos <Ten Thousand Things:Module and Mass Production in Chinese Art>,Princenton University Press, 2001, p14..

Figure 21. 아름다움 미'美', 대나무 죽'竹', 거북 귀'龜', 수풀 삼'森', 아이 밸 임'孕', 곤란한 곤'困'

대칭과 균형

: 중국 전통 미학의 특징으로 대칭의 미(美)는 중국 특유의 스타일로 높은 가치를 지닌다. 중국 문자 구성, 성어와 시사(詩詞) 외에도 건축과 일상생활용품 곳곳에서 우리는 대칭의 미를 느낄 수 있다. 비대칭의 형식 또한 자체적인 균형을 유지하려고 시도한다. 이러한 이유로 한자 형태 디자인은 비대칭 균형이라고 말할 수 있다.

입면도(elevation)의 한자 배열 (구조와 형태)

: 문자는 기본적으로 2차원 공간으로 원근감이 없고 중첩되는 형태도 매우 드물다. 고대 한자 '林'수풀 림, '竹'대나무 죽, '粟' 조 속 등이 모두 가장 직접적인 형태로 한자를 도형화한 예이다.

조감도의 한자 형태

: 높은 곳에서 내려다보는 부감법으로 주변 사물을 바라보는 것을 뜻하는데 '龜' 거북 귀, '爪' 손톱 조, '川' 내 천 등의 한자가 모두 부감 형태에 속한다.

축측도 axonometric(al) drawing의 한자 형태
: 높은 곳에서 사물이 관찰되는 방식으로 일정한 경사도로 표현된다. '森' 수풀 삼, '衆' 무리 중, '磊'돌무더기 뇌 등 한자가 이에 속한다. 위의 조감 방식과 다른 점은 들자면 관찰 대상의 규모가 더 클 때 축측 방식을 사용한다는 점이다.

투시도의 한자 형태
: 여기에서 말하는 '투시'는 중국 전통의 '투시'법으로 주관적인 투시 방법을 사용하는 것을 의미한다. 예로 '孕' 아이 밸 잉 한자를 들 수 있는데 임신부가 아이를 품은 형태를 투시하여 묘사한 것으로 우리는 중국인 특유의 구성 방식을 살펴볼 수 있을 뿐만 아니라 글자 제조자의 원형 시각과 지혜도 엿볼 수 있다.

다각도 복합
: 우리는 한자를 통해 여러 각도가 더해진 결합체라는 점도 알 수 있다. '困'곤할 곤 자가 그 예인데, 해당 글자를 자세히 분석해보면 사방을 구성한 형태는 조감하는 방식을 사용하였고, 가운데 '木'자는 입면으로 관찰한 각도를 사용한 것을 알 수 있다. 이러한 표현 방식은 가장 직접적이면서도 이상적인 구도를 활용하여 한 화면 안에 표현한 것으로서 강렬한 표현력과 생동감을 가진다.

위에서 기술한 방법은 산수화를 포함한 중국 전통 회화에서 자주 활용되는 것들이다. 동양 예술에서 활용되는 '산점散點 투시'방법과 마찬가지로 비록 이론에 불과하지만 한자의 전통 사유 양식과 관념이 다른 문자와 어떻게 구별되는지 충분히 우리에게 충분히 보여준다.

3.4.2.2 기호와 상형의 특징

e. 기호의 특징

서양에서는 일반적으로 '기호학'에 대해 다음과 같이 정의한다. "기호학은 기호를 연구하는 학문이다(Semiotics is the study of signs)." 사실 이러한 정의는 소쉬르(Saussure)에서 출발하였는데 그는 백여 년 전 '기호학' 학과를 설립하자고 제안한 인물이다. 소쉬르의 기호학 특징은 단일 기호(sign)를 'Signifier' 와 'Signified' 두 부분으로 나눈는 데에 있다. 'Signifier'는 기호의 형상을 'Signified'는 기호의 의의와 개념을 뜻한다. 이 두 부분으로 구성된 하나의 총체가 바로 기호이다.

기호학의 측면에서 살펴보면 한자는 한족 언어의 서사를 기록하는 기호 체계로서 형태, 소리, 의미의 3요소가 통합되어 있다. 하나의 한자는 글꼴의 외형과 해당 글자와 대응되는 발음 그리고 내재적 의미로 나눌 수 있다.

현대 한자의 외형은 복잡한 것에서 단순한 것으로 또 구체적 형상에서 추상적 기호로 바뀌었는데 간소화된 기호 문자는 기억과 기록에 편리하다. 그런데 한자의 'Signifier'와 'Signified'의 관계는 모호하다. 한자의 형태 체계는 대략적인 기호로 구성되며 궁극적인 내재적 의미는 이러한 기호들이 함께 작용하면서 만들어진다.

이러한 복합적 혼합 방식의 영향으로 한자로 표현하는 의미는 더욱 풍부해진다. 오늘날 다수의 한자 관련 작품들은 더이상 한자로 표현하는 뜻의 정확성만을 추구하지 않고 한자 디자인에'의미와 형태'를 담아 표현하는 것을 더 중요하게 여기며 또한 작품 안에 예술가 자신이 추구하는 본질적 의미를 담는다. 시각 기호로써 한자는 'Signifier'와 'Signified' 사이에서 만들어진 '모호'한 특성을 활용하여 다양하게 표현되는 장점을 십분 발휘할 수 있다.

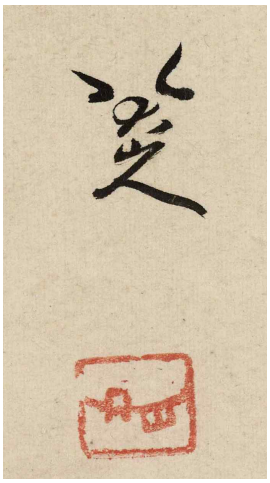


Figure 22. 팔대산인
[八大山人, Badashanren]의 서명

Figure 23. 지안핑 혜 Jianping He
중국 미술 아카데미 75주년 포스터 '예술 藝術 '

f. 상형의 특징

중국 문자는 진한 시기부터 현대에 이르기까지 '상형 문자'와 '표음 문자'가 공존하는 형성 문자 단계에 놓여 있다. 이에 상형의 특징도 한자가 만들어졌을 때부터 지금까지 줄곧 기본 특징으로 남아있는 것이다. 형태학의 측면에서 바라보면 한자의 '상형'이 바로 그 형성 과정 중에 있는 구체적인 사물에 대한 귀납, 제련, 추상, 재구성을 의미한다. '상형 특징'은 한자가 시각적 측면에서 사물의 기호를 해석하는 것으로 한자 문자도 초기의 그림을 보고 글을 익히는 것에서 상형 문자로 진화했다.

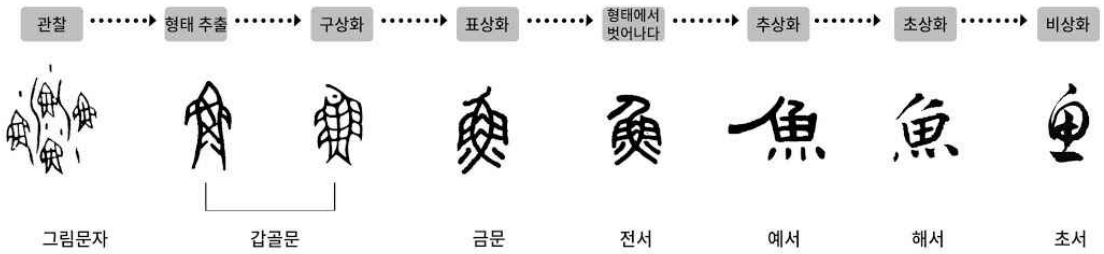


Figure 24. 한자"형 形"의 변천

3.4.2.3 표의와 은유성의 특징

g. 표의성

한자의 표의성은 한자의 발전과 맥을 같이했다. 현대 한자는 매우 추상적이고 간결하게 변모하였고 대다수의 한자가 더이상 형상과 구조로써 단어의 의미를 표현하지 않는다. 그렇다 해도 우리는 여전히 많은 예들을 통해 한자의 '표의'적 특성을 찾아볼 수 있다. '泪' 눈물 루 한자 안에는 물이 흐르는 형상이 담겨 있고, '娘'여자 낭 글자 안에는 선량한 여자라는 뜻이 담겨 있다. 이처럼 한자는 표음 문자와 달리 하나하나의 글자 안에 고유한 의미를 내포하고 있다. 이러한 표의성은 한자의 대표적인 특성 중 하나이다.

이번에는 동음자를 예로 들어 표의성을 살펴보겠다. '文' 글월 문, '蚊' 모기 문, '紋' 무늬 문, '汶' 물 이름 문, '雯' 구름무늬 문 글자는 모두 발음 나는 소리가 같다 해도 다른 형체로 기록되기 때문에 단어 의미에 있어 차이가 있다.

한자의 표의성은 구어 발음의 영향을 받지 않는다. 중국의 방언은 다양하다. 비록 발음은 다를지라도 한자를 쓰는 방법은 동일하며 말하고 나서 알아듣지 못할 때는 글자로 써서 보여주면 단번에 알아볼 수 있다.

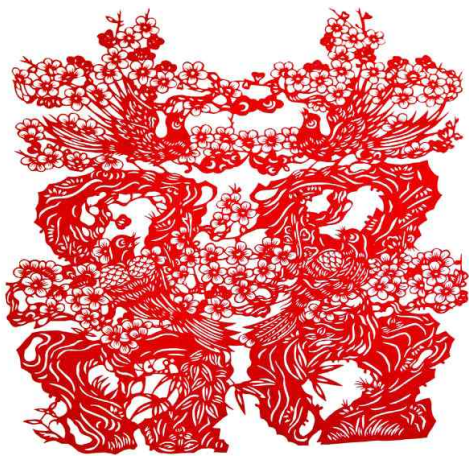


Figure 25. 중국의 전지예술(기쁠 희(喜))
쌍을 이루는 꽃(花)과 까치(喜鵲)를 사용하여 두 개의“喜+喜”기쁠 희 자를 조합한다. 한자 자전에는 두 자를 중첩한 “喜喜”자가 없지만, 중국인들은 길함을 뜻하는 요소를 한데 결합하여 “좋은 일이 두 배가 된다는” 뜻을 표현한다.

h. 은유성

한자 구성에 대해 자세한 분석을 진행하면 한자 안에 비유, 대유, 상징, 완곡 등 은유성이 포함되어 있다는 것을 알 수 있다. 회의자의 의의가 바로 여기에 있다.

선조를 뜻하는 ‘先人’이 합쳐져 늙을 로‘老’가, 알지 못한다는 뜻의 ‘不明’이 합쳐져 어두울 암 ‘暗’이, 가문을 망하게 한다는 ‘敗門’ 이 합쳐져 음탕할 표 ‘闞’자가, 부당 부정확하다는 뜻의 ‘不正’이 합쳐져 기울 왜 ‘歪’자가, 위는 적고 아래는 많다는 ‘上小下大’이 합쳐져 뾰족할 첨 ‘尖’ 등등의 회의자가 바로 돌려 표현하여 깨닫게 하는 사유 방식이 담긴 글자들이다.

그 외에도 한자로 구절이나 문장을 구성할 때 은유성의 특징도 담겨있는데 시가의 가사에서 은유성이 가장 두드러진다. 은유의 특성은 중국 문화의 ‘깨달음’ 경지를 잘 나타내주는 표현 양식이다.

4. 탐색과 실행

4.1 <ONE MOVE>

4.2 <산해경>

제4장 탐색과 실행

중국의 한자는 20세기 말이 되어서야 컴퓨터와 융합하기 시작한다. 서양에 비해 20여 년이나 늦게 출발했음에도 불구하고 컴퓨터는 빠르게 종이와 펜을 대체하였다. 이로 물리적인 과거 인쇄 시대와 달리 현대 타이포그래피 작품의 콘텐츠 또한 달라지기 시작한다. 현대 타이포그래피는 문자의 픽셀, 빛, 비트(bit), 미디어 도구 등을 작품에 담아낸다. 한자를 표현하는 범위는 인쇄 매체에 의한 종이라는 한정된 공간에서 현대적인 모니터나 스크린과 같은 동적(動的)인 공간으로 확장되었으며 또한 정적(靜的)인 한자에 시간과 공간, 스피드, 중량, 다채로운 이펙트(effect), 소리 등의 요소를 추가하여 시간적 이미지와 공간적 이미지를 동시에 표현하는 동적(動的)인 한자의 형태로 확대되면서 더 많은 의미를 표현할 수 있는 수단이 되어가고 있다.

넓은 의미에서 “움직이는” 한자를 지칭하는 명칭은 사실 매우 다양하다. 리퀴드 타이포그래피 Liquid Typography, 웹 타이포그래피 web Typography, 모션 타이포그래피 Motion Typography, 무빙 타이포그래피 Moving Typography, 다이내믹 타이포그래피 Dynamic Typography, 템퍼럴 타이포그래피 Temporal Typography, 애니메이션 타이포그래피 Animation Typography, 키네틱 타이포그래피 Kinetic Typography 등 다양한 이름들로 불리고 있다. 이러한 명칭이 의미하는 바는 상당히 유사한데, 이점이 움직이는 타이포그래피의 경계를 점점 더 모호하게 바꾸고 있다. 시대별 언어 환경 속에서 언어별로 다르게 전달되고 인용되는 것 외에 이미지와 문자의 시각적 차이로 인해 명칭을 구사하는 데에 차이가 생겨난 것이다.

현재 대다수의 움직이는 한자 작품은 모두 “TIME LINE”의 개념과 관계되어 있다. 디자이너들은 Flash, AE, PS, AI, C4D 등 도구를 사용해 영상의 개별 “프레임(frame)”을 만들어낸다. 이러한 작품의 공통적인 특징은 제한적인 시간 내에 일정한 콘텐츠를 반복 재생하는 것이다. 예를 들어 Flash 중에서 원형을 사각형으로 바꿀 때 중간에 아무리 일정한 양

의 프레임을 추가하더라도 시작점과 끝점의 결과는 고정적이며 시작과 끝의 형태도 모두 “원형”과 “사각형”이다.

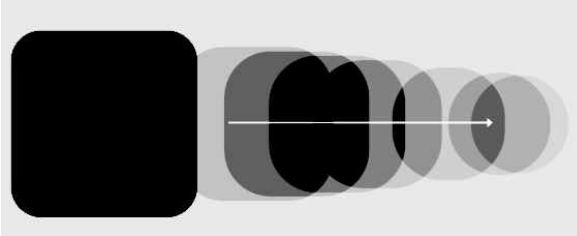


Figure 26. Flash 중의 “원형”과 “사각형”

서양의 문자는 세리프체(Serif)와 산세리프체 (Sans-Serif) 두 종류로 나뉜다. 이와 상응하는 한자를 살펴보면 명조체(明朝), 해서체(楷書), 서체(書體), 송조체(宋朝)가 전자 세리프에 해당되고 고딕활자체인 흑체(黑體)와 흑체에서 변형된 원체자(圓體)가 후자 산세리프체에 해당한다. 그림 27 <Motion Type> 은 다양한 한자 폰트는 남겨두고 거추장스러운 전통적인 요소는 다수 제거하여 전체적으로 간결하고 깔끔한 현대적 스타일에 속한다. 움직이는 키네틱 효과와 한자가 내포하고 있는 의미도 서로 연관되어 있는데 “물 수(水)”자를 표현할 때 물방울이 흐르는 효과를 사용한 것이 그 예이다. 이러한 종류의 작품 제작 기법은 “움직이는” 한자를 디자인하는 과정에서 광범위하게 사용된다.

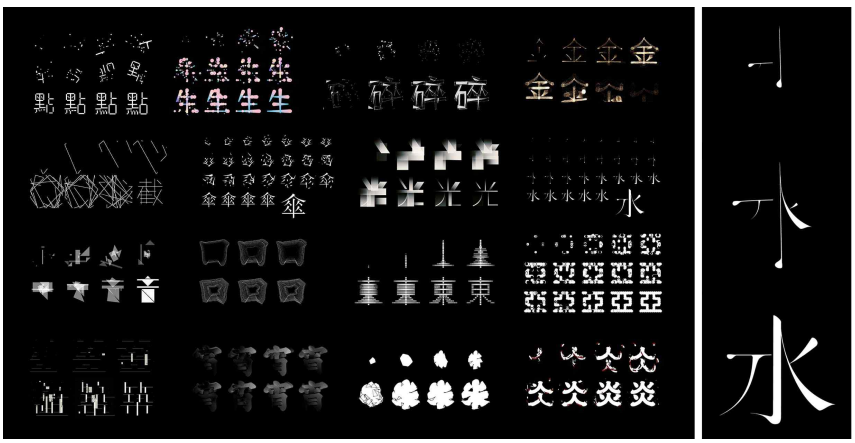


Figure 27. Ting-An Ho(何庭安) <Motion Type> [26]

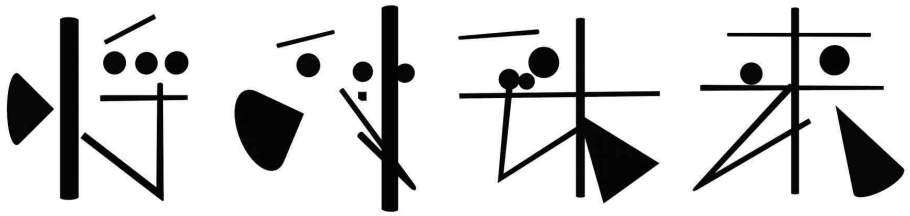


Figure 28. 井口皓泰 <將來(장래)> [27]

현대적인 감각을 살려 움직이는 한자를 디자인한 작품에는 일본의 디자이너 코우타 이구치(井口皓泰)가 창작한 <將來(장래)> GIF 애니메이션이 있다. 무한히 이어지는 길과 한자를 표현한 해당 작품에서 그는 필과 획을 분리하여 3D 공간에서 재구성하였고 애니메이션 요소를 추가하여 한자를 무한히 반복하는 형태로 만들었다.

위에서 설명한 두 작품의(그림 27, 28) 공통적인 부분은 바로 작품이 모두 “개별” 한자의 창작에 포커스를 두고 있다는 점, 미리 설정 해 놓은 콘텐츠를 고정적인 시간축 안에서 반복 재생한다는 점이다. 본 논문에서 언급하고 있는 키네틱 한자는 프로그램의 구성과 컴퓨팅 기법을 한 층 더 추구하는 경향이 있으며 작품의 최종적인 형태는 비고정적이고 시간축의 제약도 받지 않는 것이다.

일부 디자이너들이 도구와 기술의 제약에서 벗어나고자 코딩을 보조적인 도구로 삼아 작품 창작을 시도하고 있다. MIT 미디어랩의 존 마에다 (John Maeda) 는 구식 NEXT 컴퓨터에 Objective-C 언어를 사용하여 여러 프로그램을 만들었다. 초기 목적은 Adobe illustrator를 어떻게 운용할 것인지를 보여주는 것이었으나 해당 실험 작품들 안에는 키네틱 한자를 탐색하는 것으로 보이는 요소가 상당히 많았다.

---

26) https://tinganho.info/Motion-Type-Project
27) https://blog.logo123.net/23810

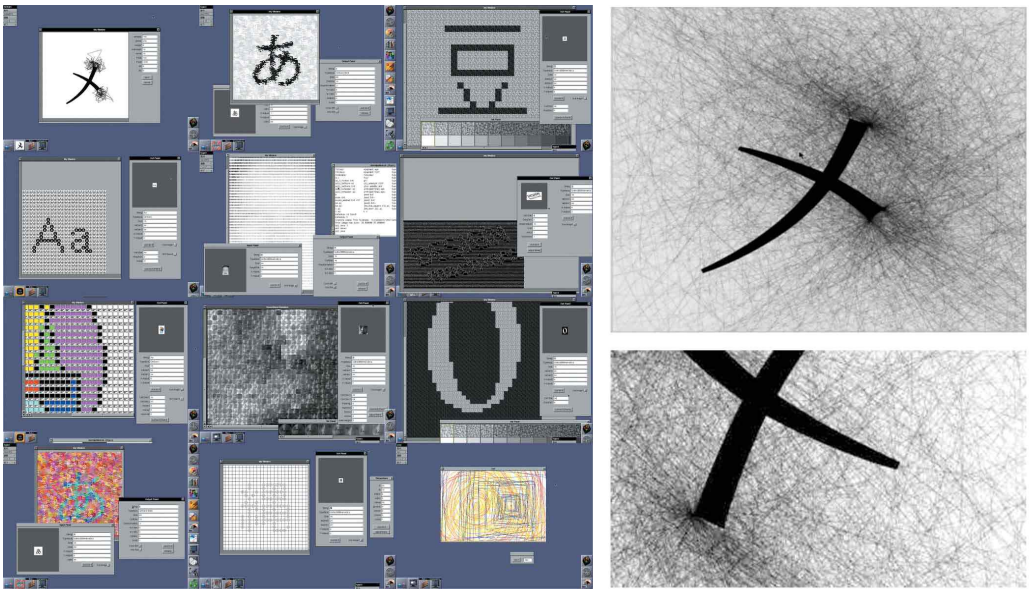


Figure 29. 존 마에다John Maeda <Illustrandom (1993)> [28]

존 마에다는 1993년에 이를 직접 처리할 수 있는 운영체제를 만들었는데 이는 그림29에 제시되어 있다. 해당 운영체제 안에는 여러 제어 프로그램이 있어 테스트 결과를 실시간으로 얻을 수 있었다. 인터랙션(interaction)이 가능한 해당 디자인은 현재에 보기에도 매우 앞선 것이다. 존 마에다의 탐색과 연구에 힘입어 서양에는 날로 우수한 키네틱 타이포그래피 작품들이 쏟아졌다. 하지만 진정하게 한자와 연관되었다고 할만한 키네틱 디자인은 매우 드물었다. 이러한 까닭은 사실 한자 자체적인 특수성과 깊은 관련이 있다. 한자는 동방 문화의 정수로서 복잡한 구조로 인해 학습이 쉽지 않을 뿐만 아니라 종종 한 글자 안에 여러 의미를 압축하여 포함하는 특징을 갖고 있어 한자 문화권에서 오래 생활하지 않은 환경에서 이를 구사하기란 매우 어렵다.

궈루이원(Raven Kwok)은 중국의 프로그래머이자 시각 디자이너로, 그의 예술 창작과 학술 연구 분야는 '시각 미학으로 표현 가능한 컴퓨터 프로그래밍 알고리즘 '이다. 2019년에는 컴퓨터와 알고리즘을 사용하여 아티스트 Jia의 작품 < The Chinese Version >을 새롭게 재구성하였다.

28) https://www.youtube.com/watch?v=IyN8Ga3CdQU

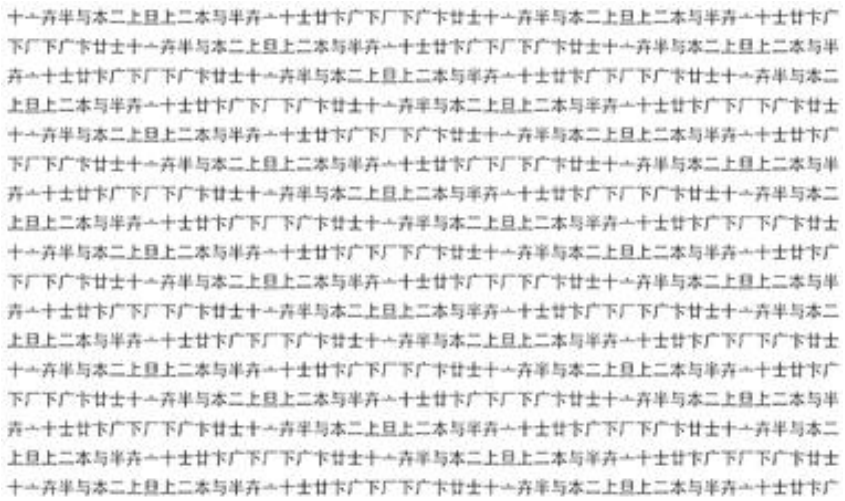


Figure 30. “The Chinese Version 001: Untitled”, acrylic on canvas, 3000×1900mm, 2012

그림 30는 아티스트 Jia가 한자 “필획” 의 특징을 활용하여 창작한 작품이다. 그녀는 “가로 필획”이 포함된 문자를 고른 다음 해당 문자들을 나란히 배열했다. 배열한 문자 사이에는 관련성도 없고 문장을 구성하지도 않지만, 시각적으로 질서정연한 물결 선을 이루고 있다. 얼핏 간단해 보이는 작품 창작의 이면에는 한자의 특징을 오랜 기간 연구하고 해당 문화의 특징을 심도 있게 이해하는 작가의 시선이 담겨있다. 궈루이원은 코딩을 창작의 붓으로 삼아 컴퓨터를 활용하여 한중일 통합 한자(CJK Unified Ideographs) 배열을 진행하고, 한자 필획 “가로선”의 높낮이를 활용하여 물결치는 선을 구성하였다. 궈루이원의 물결치는(그림31) 선은 시작점과 마지막 점이 없고 실시간 변화하는데 이 점이 Jia의 작품과 구별된다.

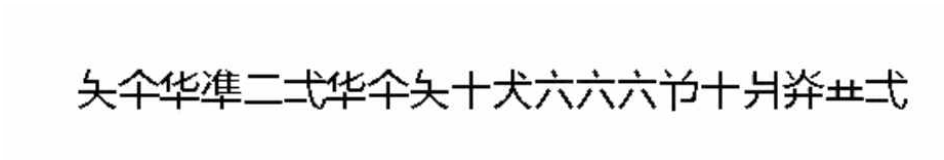


Figure 31. 궈루이원 (郭 鋭文)

현재 키네틱 한자 작품의 수량은 많은 편이 아니다. 하지만 점점 더 많은 수의 아시아 디자이너들이 한자의 예술적 가치를 주목하고 작품 창작에 기술을 융합하고 있는 점은 매우 고무적인 일이다. 위에서 언급한 것처럼 움직이는 한자를 정의하는 경계는 모호하고 그것을 지칭하는 명칭도 다양하다. 본 논문에서 언급하는 키네틱 한자 범위는 한층 더 포괄적일 수 있다. 키네틱(Kinetic)은 물체의 움직임이나 변화, 실행 또는 정렬하는 것을 의미한다. 이에 따라 광의적으로 “키네틱 한자”는 시간과 공간 안에서의 타이포그래피의 변화, 실행, 정렬 등을 포함한 다양한 움직임을 디자인 하는 것을 의미하며 모션 중의 “프레임”과 “시간축”의 개념도 가지고 있을 뿐만 아니라 앞으로는 실시간으로 생성(Generative)되거나 양방향으로 인터랙션(interactive)도 가능할 것으로 보여진다.

한자 키네틱 타이포그래피 작품의 창작을 위해서는 이론 분석을 진행하는 동시에 여러 실행 과정을 통해 새로운 디자인 기법을 고안하고 검증할 필요가 있다. 이에 4장에서는 본인이 실제 작업한 프로젝트를 소개하고자 한다. 해당 프로젝트는 담당 교수님의 지도 아래 수업 틈틈이 완성한 과제로서 다른 디자이너들의 빼어난 키네틱 타이포그래피 작품과 비교하면 부족한 점이 많지만 프로젝트 작품에 대한 정리와 분석을 진행하는 작업은 본 연구자의 작품 창작 노하우와 스타일 구성에 적지 않은 도움이 되었기에 다음과 같이 제시한다.

아래에 제시하는 프로젝트는 총 2건이다. 두 작품의 창작 배경과 다루는 주제는 상이하지만 해당 작품들은 모두 프로그래밍 언어를 직접 사용하여 제작한 것들이다. 첫 번째 <ONE MOVE> 작품은 디지털 기호와 생성(Generative) 그래픽을 결합한 것이고, 두 번째 작품<산해경.>은 한자 부호와 영상을 결합한 것으로 주로 “한자”의 응용에 집중되어 있다.

4.1. <ONE MOVE>

<ONE MOVE>는 다양한 국적의 디자이너 3명으로 구성된 팀 프로젝트로 문화적 배경이 서로 다른 디자이너들이 서울에서 함께 모여 작품을 진행하였다. 이런 연유로 프로젝트 진행 초기에 가장 먼저 모든 구성원이 공통으로 관심을 가지는 주제를 정할 필요가 있었고 우리는 서울과 세계 다른 도시의 문화적 차이점에 주목하였다. 흥미로운 점은 구성원 모두 서울을 오가는 용달차에 대해 관심이 많다는 점이었다. 한국의 도시에서 흔히 볼 수 있는 용달차는 국민들이 애용하는 저렴한 운송 서비스 수단으로 보통 이사할 때 많이 사용된다. 도로와 골목을 오가는 용달차 안에는 가전 용품, 가구, 생활용품이 실려 있었는데 외부에 노출되어 있어 누구나 볼 수 있게 전시되어 있었다. 이삿짐 용달차는 하나의 소형 전시 쇼윈도와 같이 한국인들의 일상을 고스란히 보여주었다.

Figure 32. 이미지, 이삿짐이 가득 실려있는 용달차

한국에서 도시에 거주하는 사람들의 이사 빈도는 매우 높은 편이다. 아마도 현대적인 도시 서비스가 잘 갖춰져 있어 이사가 편리하고 신속하

며 비용도 저렴하기 때문일 것이다. 또는 도시에 거주하는 사람들이 다변화하는 생존 환경, 스트레스, 생활 수준 차이 등의 이유로 마주해야 하는 하나의 대응책이었을 수도 있다. 그러나 어떠한 이유에서든지 이렇게 이사하는 방식이나 이삿짐 그리고 이사할 때 사용하는 수단은 모두 한국 고유의 특색을 자아내기에 충분했다. 프로젝트 구성원들은 모두 각자의 특기와 재능을 활용하여 서울의 도시가 한층 더 풍부하고 다채로워지고 또 한국에서 생활하는 사람들에게 편리함을 제공하기를 바랐다.

프로젝트의 최종 목표는 한국 이사 문화와 관련된 시각 이미지 구조를 구축하는 것이다. 시각 이미지 구출을 위해 이삿짐을 싣고 오가는 용달차를 보며 느꼈던 바를 먼저 소개하겠다.

서울에서 1.5톤 이삿짐 트럭에 가득 실린 짐을 보는 것은 어느덧 우리 삶의 부분적 풍경이 되었다. 이사는 무엇을 의미하는가? 이사를 한다는 것은 '새로움'이라는 긍정적 모습을 내포하기도 하지만 집이 없는 사람에게는 때론 무한한 고단함을 의미하기도 한다.

매년 계약이 끝날 시점 집세가 올라가지 않을까? 어쩔 수 없이 이사한다면 어디로 가는 게 좋을까? 적당한 가격의 집을 찾을 수 있을까? 이삿짐은 어떻게 싸고 이동하는 게 좋을까? 등 다양한 고민을 하게 된다. 이렇게 이삿짐 트럭에 드러내 어진 다양한 가구와 집기는 우리 서민들의 일상화된 고충을 드러내는 것이다.

이렇게 이사하는 횟수가 잦을수록, '새로움'이라는 희망보다는 '괴로움'이라는 걱정거리의 횟수가 늘어가기만 한다. 세상에서 가장 편안한 버팀목으로 되어야 할 공간, 그리고 고단하지만, 행복을 느껴야 할 이사 준비과정은 어느덧 서민들의 깊은 '고충'을 적나라하게 드러낸다.

그렇다면 때마다 철새처럼 이동하는 우리의 '이사 풍경'을 긍정적으로 개선해 볼 방법이 없을지 고민해 본다!

< 2017 감상글>

우리는 이사할 때 가장 먼저 모든 물품에 대해 분류하기 시작한다. 물건을 효과적으로 넣고 물품을 구분하는 것은 중요한 문제이기 때문이다. 시각 이미지 구축을 위한 첫 번째 단계도 물품 분류에서 시작하였다. 이사하는 사람의 물건 종류와 수량이 각기 다르기에 프로젝트 팀은 상자의 크기를 임의 대로 정하고, 3가지 색상을 정하여 고객이 자유롭게 선택하

여 사용하도록 했다.

상자의 크기도 자유롭게 바꿀 수 있고, 대응되는 색상도 고객의 니즈에 맞추어 결정할 수 있어서 매칭되는 조합의 수는 매우 다양해진다. 이러한 무작위적인 랜덤 속성을 표현하기 위해 기존의 전통적인 디자인 방식만을 사용하면 매우 복잡하고 번거로워진다. 이러한 이유로 프로세싱(processing) 프로그램을 사용하여 디자인을 보조하고, 알고리즘과 랜덤 순환 방식을 통해 다양한 변화를 주었다.

프로젝트 메인 이미지 중의 logo 디자인은 가변적인 디자인 기법을 사용하였는데 이를 통해 이동, 분류, 크기 등 특수한 상황을 더 잘 표현할 수 있었다.

위에서 언급한 최종 시각 이미지 효과는 아래의 그림에 제시되어 있다.

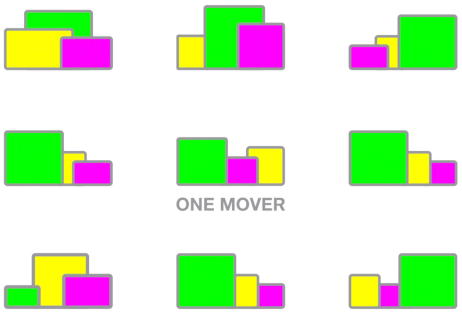


Figure 33. 브랜드 logo

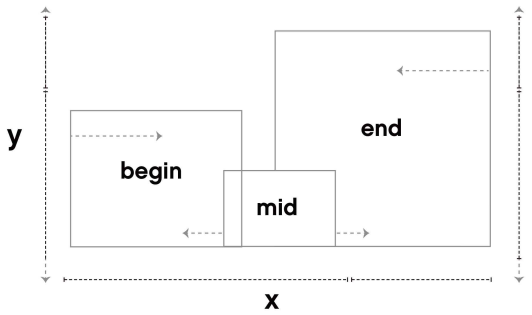


Figure 34. 제작원리

용달차 한 대에 적재할 수 있는 화물의 공간(x, y)는 고정되어 바뀌지 않고, 3종 색상의 상자 크기는 바뀔 수 있는 조건 하에서 우리는 다양한 배열의 상자 조합을 만들 수 있다. 이에 움직이는 logo의 프로그래밍 logic은 먼저 상자에 begin, mid, end의 3가지 상태를 부여한다. 가령 상자 한 개를 좌측에서 우측으로 움직일 때 30프레임이 필요하다고 가정한다면, begin이 가장 좌측의 상태이고, end는 가장 우측 상태이며, 각각의 프레임 안의 상태는 바로 mid가 된다. 프로그램 시행 시에 랜덤으로 표시 순서(.shuffle)를 바꿀 수 있고, 초기 상태에 있는 상자의 좌우 위치를

판단하여 중간값 mid를 계산해낼 수 있다.

이렇게 상호 교차하여 랜덤으로 생성되는 평행이동 방식은 “이사”의 테마에 가장 적합하고, 계산 후에 만들어진 다량의 그래픽은 또한 인쇄 제품에서 매우 강렬한 시각적인 효과를 자아낸다. 이외에도 야간 운송의 편리함을 위해 상자 모서리마다 빛 반사 재질(3M FABRICS TRIM: 원단 백킹에 마이크로 글라스 비드가 부착된 봉제용 반사 소재)을 부착하였는데, 현장 전시도 35)에 실물 상자, 포스터, 담요 등이 있으니 참고하길 바란다.

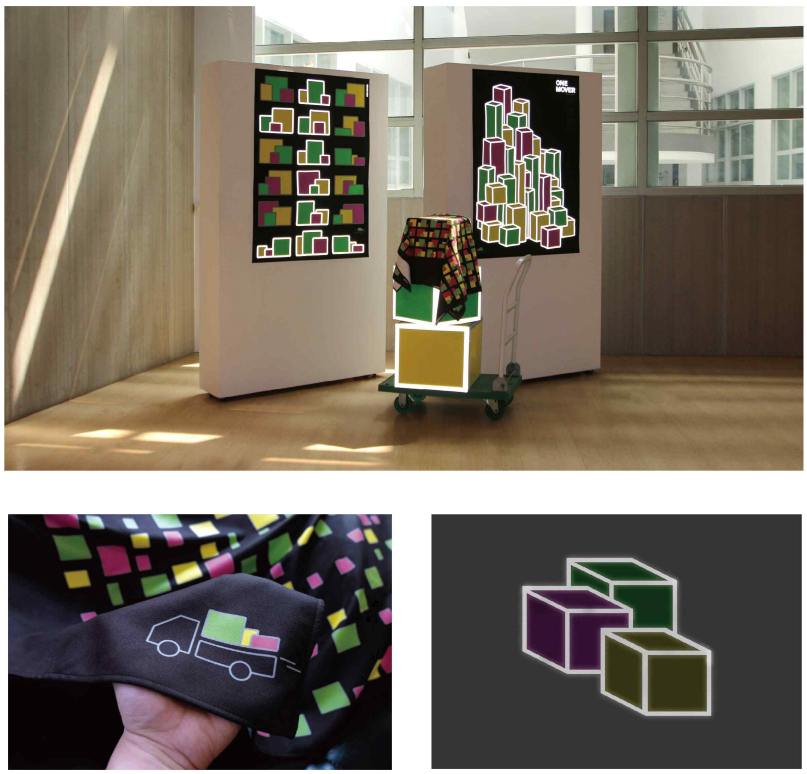


Figure 35. <ONE MOVE> 전시 현장도

정적인 상태에 있는 인쇄 제품에 그래픽과 문자 형태를 결합한 케이스는 매우 많지만, 계속 변화하는 logo를 사용해 브랜드의 의미를 표현하는 작업은 제약 받기가 쉽다. 더 많은 대중의 시선을 사로잡기 위해 본 연구자는 ONE MOVE 형태를 접목하여 선명한 색채, 랜덤 조합, 이동

빈도 등의 특징을 바탕으로 스크린 기반의 키네틱 타이포그래피 포스터를 창작하였다. 포스터의 제작 방법은 다음과 같다. 먼저 사전에 숫자 “1”의 형태를 설정하고, 구체적인 위치 좌표를 산출하여 최종적으로 결과를 행렬 안에 저장하였다.

스크린 안의 숫자 “1”의 비트맵 정보를 2진법에서 6진법으로 전환하여 산출하기 위해 해당 포스터에 PCtoLCD2002 프로그램을 사용했다. 그 다음 프로세싱을 사용해 해당 정보들을 매칭하고 그래픽의 그라데이션(gradation), 페이드 아웃(fade-out), 위치 이동 등 효과를 전체적으로 조절했다.

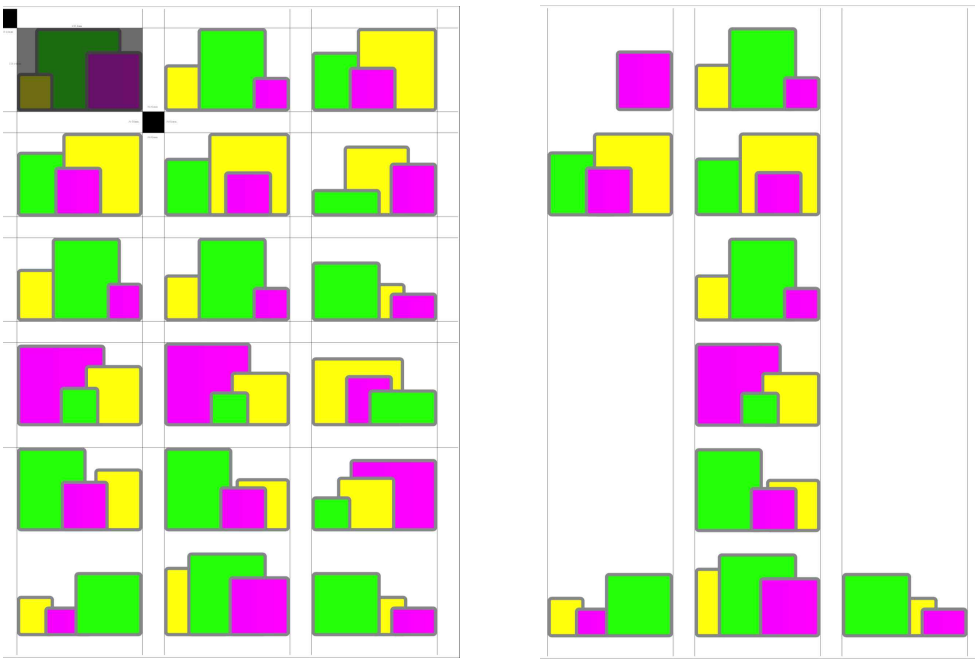


Figure 36. 선행 설계하여 표시한 위치

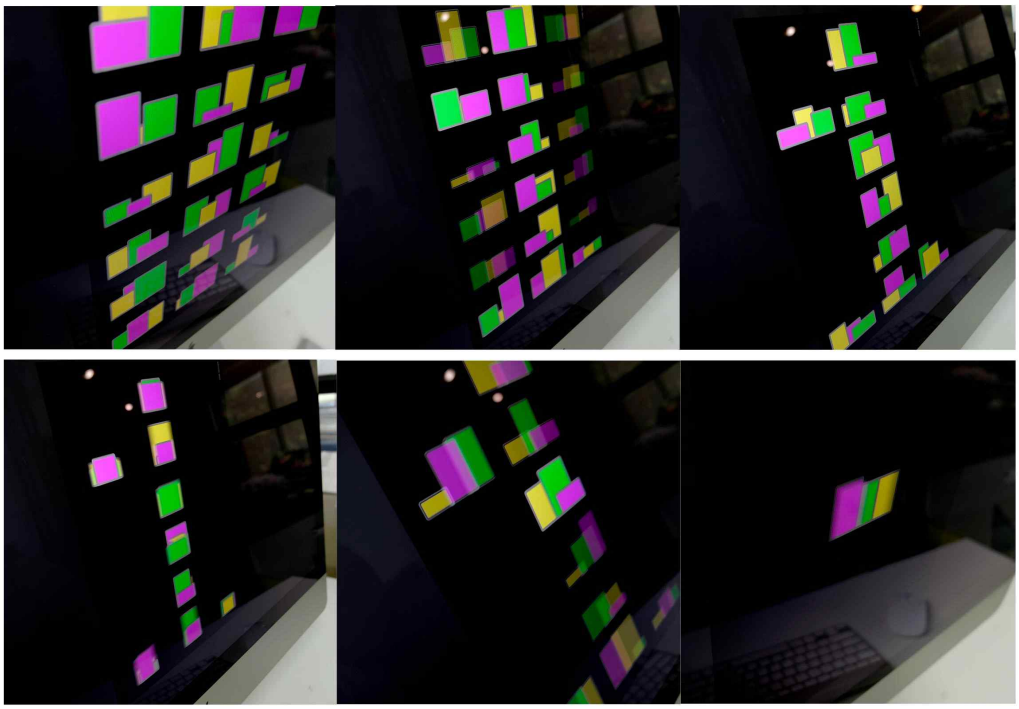


Figure 37. 현장에 전시한 운행되고 있는 모니터
www.xinlin.art - /2018/1 , lxl.snu.ac.kr - /2018/1

4.2 <산해경>

<산해경> 작품의 영감은 본 연구자가 서울에서 관람한 두 전시작에서 유래하였다. 2017년 안상수 교수의 <날개.파티> 전시회에서 우연히 보게 된 <도자기 타일> 도자기에 잉크(그림 38) 작품은 네모난 도자기와 한글 요소의 형식을 사용한 또 다른 작품 2015년 타이포잔치 김두섭 교수의 <아파타입> [29] 을 연상시켰다. 두 교수의 작품에 사용한 물리적 매개인 “도자기”는 전통 인쇄 매체와 구별되며 웅장한 작품을 완성하기 위해 대량의 인적, 물적 자원이 투입되었다는 점을 잘 보여준다.

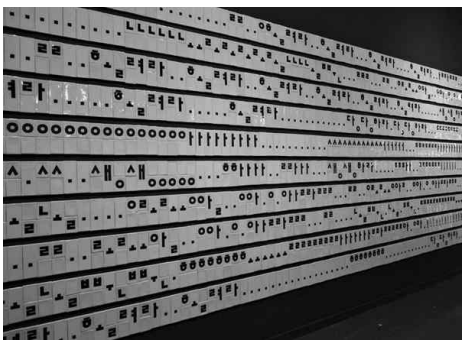


Figure 38. 안상수 <도자기 타일> 도자기에 잉크, 1000x300cm ,2017

Figure 39. 김두섭 <아파타입> , Mixed media 332.5x695.5cm, 2015

만일 이들 작품이 정적인 상태에만 국한되지 않는다고 가정한다면 작품은 시간이 흐르면서 계속해서 이동하며 움직일 것이다. 마치 김포공항의 미디어아트 ‘달 항아리’ (그림 40) 처럼 말이다. 그렇다면 이러한 작품은 관객에게 또 다른 체험이 될 수 있지 않겠는가?
이에 <산해경.>의 창작 취지를 아래와 같이 설정하였다.

1. 고효율, 저비용 1인 작품
2. 중국 전통문화와 관련된 작품
3. 한자 타이포그래피에 기반한 작품
4. 스크린에 기반하여 움직이는 뉴미디어 작품

---

29) 타이포잔치 2015 . http://typojanchi.org/2015/ko/1-doosup-kim

Figure 40. 김포공항 <달 항아리> 30)

작품을 <산해경>이라 지은 까닭은 중국에서는 “많은” 사람을 형용할 때 보통 바다 해 “海”자를 사용하여 표현하기 때문이다. 예를 들어 바다와 같은 양을 뜻하는 해량 “海量”자처럼 한자로 많은 수를 표현할 때는 통상적으로 바다 해 “海” 자나 “海” 자가 들어간 단어를 사용하여 형용하는 경향이 있다. 두 번째 이유는 아시아 전통문화에서 “자연”은 영원히 변치 않는 창작 소재로서 물 수 “水”자가 있으면 뫼 산 “山”자가 꼭 있게 마련인데, 고금을 막론하고 다수의 문인들이 산수화 같은 예술적 경지를 추구했기 때문이다. 마지막 이유는 저서『산해경』이 중국에서 신화를 가장 많이 기재한 고전이자, 지리 분야의 백과사전으로 디자이너들이 배울 부분이 매우 많기 때문이다. 전통문화를 사랑하는 중국 디자이너들은 모두 산해경을 재창작하는데 책 디자인과 편집디자인뿐만 아니라 삽화 디자인 등 분야에 우수한 작품이 매우 많다.

앞서 언급한 것처럼 중국에서는 현대 한자를 네모난 글자 “方塊字”라

30) 매일경제 www.mk.co.kr/news/society/view/2017/10/677234/

고도 부르는데, 이“方塊字”은 위의 “네모난 도자기” 안에 있는 한글처럼 고정 비율의 “사각 테두리” 안에서 완성된다. 한자 중 서체 필획의 굵기(그림 41), 필획 밀도(그림 42)의 차이점은 모두 시각적으로 명암의 차이를 만들 수 있다는 점이다. 현대의 디스플레이 방식은 모두 디지털에 기반하여 만들어졌는데, 절묘하게도 한자의 “필획”과 같이 디지털 디스플레이 안에도 “화소”라는 최소 단위가 있다. 컴퓨터의 최소 단위 (화소)와 한자의 최소 단위(필획)을 결합하는 것이 바로 <산해경> 작품의 핵심 디자인 원리이다.

Figure 41. 신해 서체, 믿을 신“信”자의 필획 굵기를 배열.

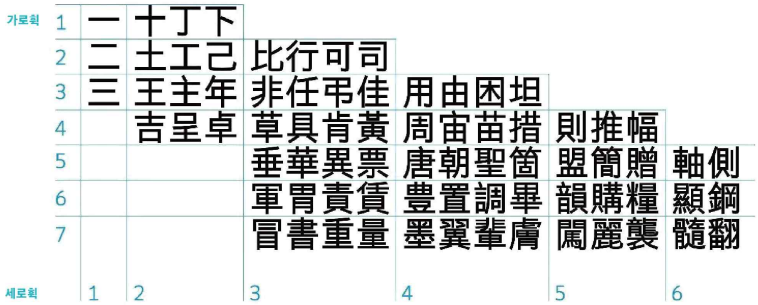


Figure 42. 한자 필획의 조밀도 배열이 두드러짐

그림 42 에서는 “신해 信黑 xjf” 서체만을 사용하여 배열을 진행하였다. 목적은 서체가 달라 발생하는 효과의 차이를 배제하기 위해서이다. 이는 또한 필획의 차이만으로도 한자에 다양한 밀도를 만들어 낼 수 있는 점도 보여주는데 해당 속성으로 시각적인 명암, 깊이, 변화 등의 여러 효과를 줄 수 있다.

<산해경>은 프로세싱을 통해 그래픽 요소 하나하나를 한자로 바꾼 방식으로 제작되었다. (r+g+b)/3으로 산출한 회색의 평균값은 그레이 스케일(gray scale) 값이 0~255이어서 해당 작품을 25개 등급으로 나누었다. 그레이 스케일 값과 필획 수가 대응되면 그레이 스케일 값은 점점 많아지고 필획 수는 점점 적어진다. 구체적으로 대응하는 글자는 해당 요소의 그레이 스케일 값에 의해 결정되고, 색채와 대응되는 한자의 그레이

스케일 값은 컴퓨터의 그레이 스케일 값에 따라 문자열을 읽어낸다. 그 외에도 최종적으로 움직이는 시각 효과를 만들어 내기 위해 움직이지 않는 사진을 로딩해 GIF나 영상 파일로 바꾸었다. 그림 43은 프로젝터를 사용하여 진행한 작품을 전시한 것이다.

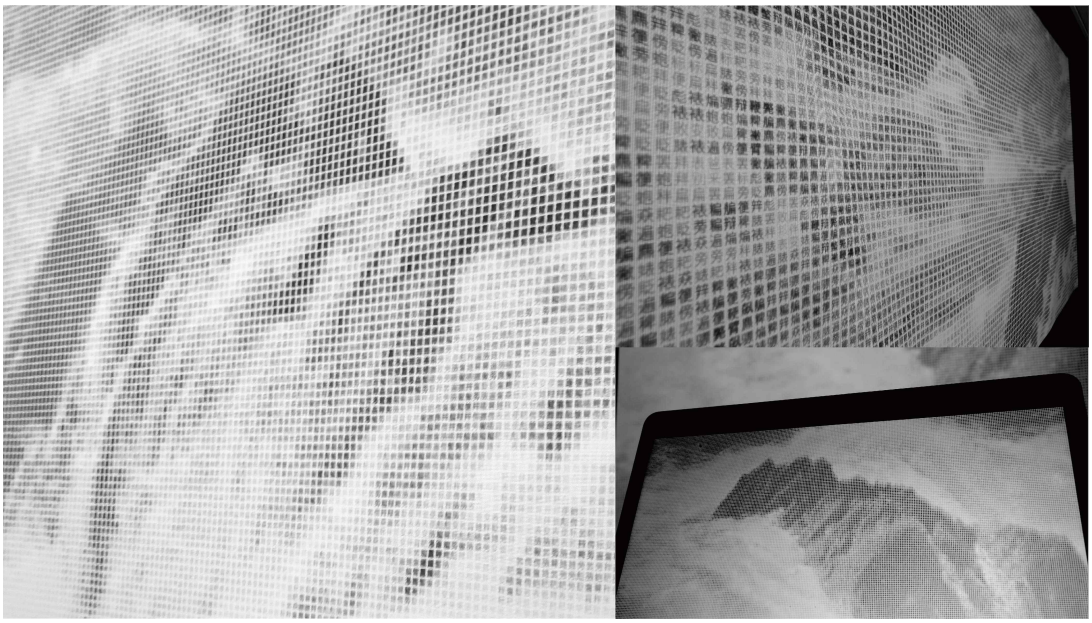


Figure 43. <산해경.> 현장 사진
www.xinlin.art - /2018/2 , lxl.snu.ac.kr - /2018/2

한자는 구조, 쓰는 방식, 읽는 방식, 내포하고 있는 의미 등에 있어 다른 문자와 많은 차이가 있다. 이러한 한자의 특징을 모색하기 위해 이를 잘 드러낼 수 있는 기술 수단을 선택하여 한자의 특징을 보여주는 작업은 매우 흥미롭고 재미있는 탐색 과정이 아닐 수 없다. 키네틱 한자 창작 과정에서 시각 이미지로 표현하는 일도 물론 중요하지만 한자가 내포하고 있는 정보도 함께 담아 진행하는 창작이야말로 한자 고유의 매력을 더 잘 보여줄 수 있는 디자인이라 생각한다. 동시에 이러한 점은 제 5장에 제시되어 있는 작품을 창작할 때 본 연구자가 반복해서 고민해야만 했던 어려움이기도 하다.

5. 작품 제작

5.1 <甲骨文：鳥>：**갑골문：새**

5.2 <蜂擁而至>：**봉옹이지**

5.3 <大音希聲, 大道無形>：**대음희성 대상무형**

5.4 <弱肉强食>：**약육강식**

5.5 <風回路轉>：**풍회로전**

5.6 <以大化小 , 以小見大>：**큰 것, 작은 것**

5.7 <你中有我, 我中有你>：**너 , 나**

5.8 <言多必失>：**말은 실수를 낳는다**

5.9 <合二爲一>：**둘이 모여 하나가 되다**

5.10 <一 木>：**나무 한 그루**

5.11 <一 草>：**풀 한 포기**

5.12 <一 花>：**꽃 한 떨기**

5.13 <冬節>：**동절**

5.14 <人心>：**인간의 마음**

5.15 <看到風>：**바람을 바라보다**

5.16 < 無 >：**무**

제5장 작품 제작

키네틱 한자 디자인은 글꼴 디자인과 이러한 글꼴을 사용하여 진행하는 타이포그래피 디자인으로 나뉜다. 3장에서 진행한 분석을 토대로 우리는 한자 고유의 특징을 활용하여 단일 '글자(부분)'에 여러 종류의 '폰트(전체)'를 조합하는 방식으로 제3의 디자인 방식을 창작해낼 수 있다. 5장에는 키네틱 한자 디자인의 기본적인 특징을 보이는 작품을 살펴보며 이를 3가지 유형으로 나누어 디자인한 내용이 담겨있다.

5.1 <甲骨文 : 鳥> : 갑골문 : 새

현재 세계에서 통용되는 문자의 대다수는 표음문자이다. 표음문자는 수십 개의 자음과 모음을 사용하여 글자의 말소리를 기록한다. 이와 달리 한자는 표의문자로 '소리'와 '형태'을 하나의 글자 안에 담는다. 초기 발전 단계의 한자는 그림문자로 '형상'은 그림문자에서 시작되어 발전되었다. 문자와 관련된 연구를 통해 우리는 해당 문자의 기원과 발전 역사를 알 수 있다. 상대(商代) 갑골문은 현재까지 발견된 문자 중 가장 오래된 한자이다. 고대인들은 원시 상태에서 대자연을 기록하기 위해 시각기호를 창조해냈다. 갑골문의 그림 하나하나에는 대자연을 바라보는 고대인들의 호기심과 원형에 대한 인식이 담겨 있다.

한자는 변천을 거치며 오늘날에는 매우 추상적이고 간결한 문자로 바뀌었다. 한자에 담긴 가공되지 않은 원시정보를 복원하는 일은 시각적인 즐거움을 전달하는 매우 특별한 일이 아닐 수 없다. 현대 한자의 '상형성'은 이미 많이 퇴화되었지만, 갑골문은 자연계의 동식물에 대한 형태를 직접 묘사하고 있다. 이러한 그림의 형태에 다시 생명력을 불어넣는다면 한자의 무한한 상상력을 대중에게 다시 재현해낼 수 있을 것이다.

한자의 '상형성' 특징을 전달하기 위해 본 연구자는 첫 번째 작품의 주제를 "갑골문 : 새"로 정하였고 조류의 원시 행위를 시뮬레이션 하여 문

자에 생명력을 부여하였다. 자연계의 모든 생명체는 하나의 독립된 개체로서 주변 환경을 느끼고 인지한다. 취약한 개체는 때로 생존율을 높이기 위해 방대하고 복잡한 시스템(Complex systems)을 형성하는데 이 시스템을 자율 지능체(Autonomous Agents)라고도 부른다. 자연계 동물의 군집 행위를 모방하기 위해 1980년대 컴퓨터 과학자 Craig Reynolds가 생명체 행위를 계산하는 알고리즘을 발명했다. 가장 대표적인 예가 바로 '무리 (flocking/swarming) ' 행위의 연구에서 도입된 '보이드(boids)' 모형이다.

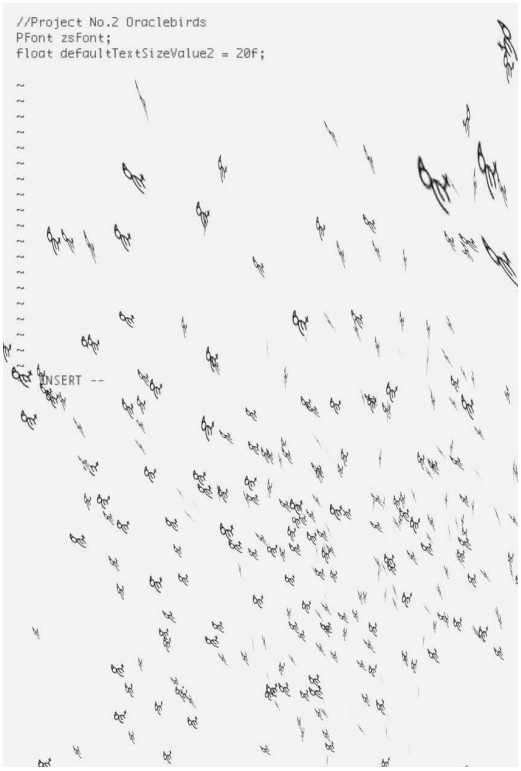
작품 <갑골문: 새> 역시 이러한 논리를 따랐으나 Kyle McDonald의 알고리즘을 사용했다. 점에서 선까지의 최소 거리와 평면 방정식 (Equation of a plane) 등을 직접 계산하여 각 점의 위치를 얻어냈다.

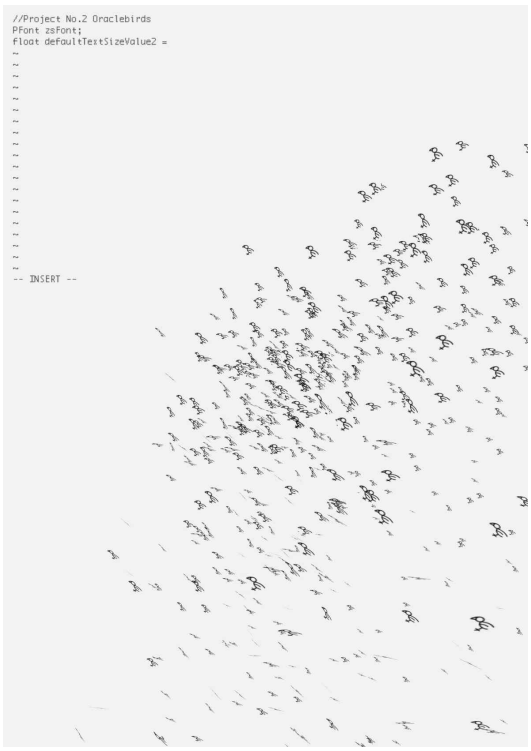
작품 감상 에세이:

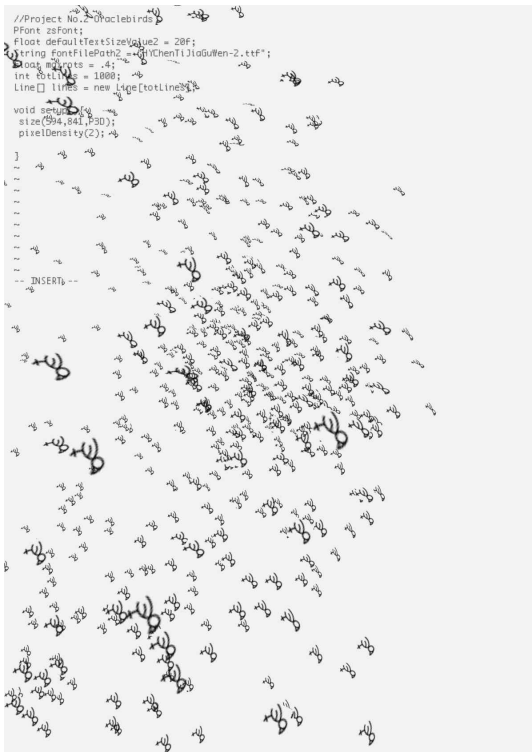
<甲骨文：鳥>　　<갑골문：새>

所有都是最初的階段　　모든 것은 다 최초의 단계
動物本能的最初階段　　동물 본능의 최초 단계
人類文明的最初階段　　인류 문명의 최초 단계
漢字發展的最初階段　　한자 발전의 최초 단계
我的作品的最初階段　　내 작품의 최초 단계

<Oracle: Bird>
All in its infancy
The first stage of animal instinct
The first stage of human civilization
The initial stage of Chinese characters
The first stage of my work







☑다차원 ☑모듈화 ☑기호화 ☑상형성 ☑표의성 ☑은유성

5.2 <蜂擁而至> : 봉옹이지

문자 자체는 정보를 가장 효율적으로 전달하는 매개체이다. '상형'의 특징을 갖고 있는 갑골문이 작품의 대상이 될 때 전달 메시지가 너무 직접적이고 간단하면 대중은 흥미를 잃어버리기 쉽다.

첫 번째 작품 스타일을 이어나가기 위해 시리즈 작품으로 <蜂擁而至 봉옹이지> 를 창작하였다. 해당 작품은 한자 성어로 벌떼처럼 밀려들다는 뜻을 가진다. 많은 사람들이 우르르 한곳을 향해 모이는 것을 형용한다. 작품에서 무리를 따른다는 뜻의 "從衆"두 자를 조합하여 자연계 곤충의 집단 행위를 그려냈다. 그리고 한자가 가지고 있는 '표의성'과 '은유성'으로 작품의 의미를 풍부하게 표현하였다.

우리는 많은 동물들이 유사한 군집 행위를 한다는 것을 익히 알고 있다. 고기떼, 흰개미, 꿀벌, 무리를 짓는 소 등을 비롯하여 심지어 공공장소의 인간 무리까지 모두 군집 행위를 한다. 인류 역시 원시 형태의 '군중'심리를 가지고 있다. 원시사회에서 무리는 인류의 생존율을 높였다. 인류가 발전을 거치면서 오늘날에는 '군중을 따르는 從衆'과 '생존'이 더 이상 직접적인 관계가 없는 것처럼 보인다. 그러나 본 연구자는 인간의 마음을 어지럽히는 것처럼 보이는 군집 행위가 사실 현대 사회의 중요한 생존 법칙인 "중용"이라고 생각한다.

<蜂擁而至 봉옹이지>는 FlowField 기법을 바탕으로 군집 행위(Group Behaviors)의 개념을 더한 작품이다. FlowField의 물체를 따라가는 가운데 Flocking 환경 판단 기능을 넣었다.

작품 감상 에세이:

<蜂擁而至>

如今的“從衆”代表着缺乏个性和缺少突破
“從衆”在人們心中就如同這一團團雜亂的飛虫
而正是這看似平庸的生存方式
才讓脆弱的人們能感悟中庸的生存之道

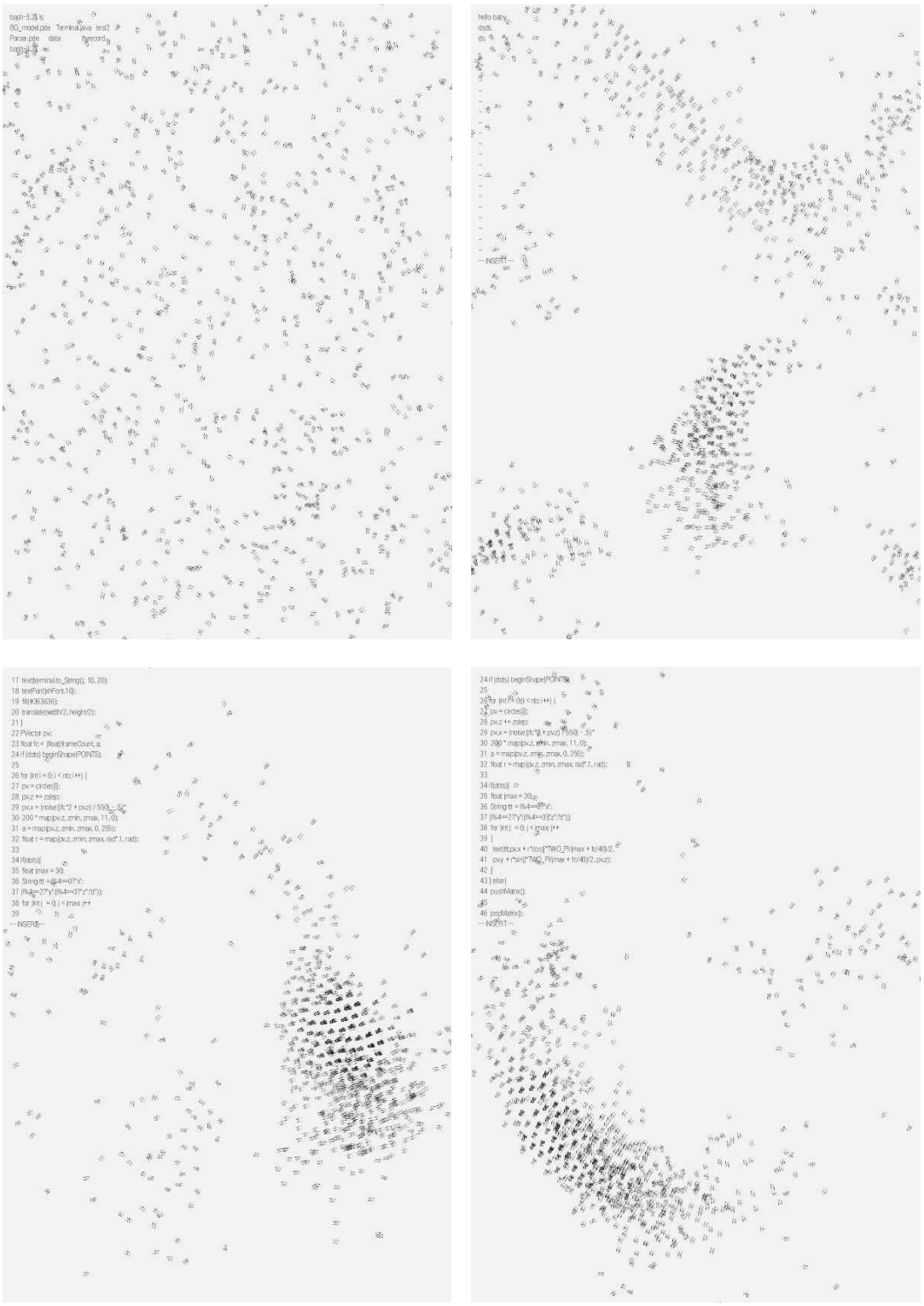
<봉옹이지>

오늘날 ‘從衆’은 개성 결핍과 도전 부족을 대표한다
‘從衆’는 인간의 마음을 어지럽히며 나는 곤충과도 같다
평범한 것처럼 보이는 생존 방식을 취해야만
연약한 사람들은 중용의 생존 이치를 깨달을 수 있다

* 從衆: 무리를 좇다

<Swarming>

Today, conformity suggests a lack of
characteristics and breakthroughs.
Conformity is just like a mess of buzz.
It may seem a mediocre way of life.
At the same time, it enables vulnerable
to have a taste of the golden mean.



☑다차원 ☑모듈화 ☑기호화 ☑상형성 ☑표의성 ☑은유성

5.3 <大音希聲, 大道無形> : **대음희성 대상무형**

자연은 매우 복잡하면서도 신비롭다. 인간은 물리, 화학, 생물 등 분야를 학습하며 각각의 자연현상을 이해해왔다. 오늘날에는 소프트웨어 기술로 자연을 시뮬레이션 하는 연구 방식이 매우 발전했다. 작품 안의 '나'는 자연 세계를 온전히 복제하려고 하지 않는다. 앞의 시리즈 작품을 이어나가기 위해 본 작품은 수중 생물 행동을 시뮬레이션 하였고 더불어 '물'에서 생성된 효과에 포커스를 두고 그것을 강조하였다. 왜냐하면 '물'은 중국 문화에서 매우 중요한 위치를 점하기 때문이다. '물'은 '도덕', '경지', '무형' 등 전통 가치관과 자주 연관된다.

작품의 제목은 노자의 『도덕경』 중의 "大音希聲, 大道無形"에서 유래했다. 이는 큰 소리는 오히려 들리지 않고, 큰 모양은 오히려 형태가 없다는 뜻이다. 물은 무색 無色, 무미 無味, 무성 無聲, 무형 無形이다. 이러한 상태로 세상에 존재하더라도 물은 어디에나 있으며 모든 생명체에 없어서는 안 될 만물을 포용하는 존재이다.

작품은 한자로 수중 생물의 형태를 시뮬레이션 하고 2D WATER 알고리즘을 사용하여 물의 물결무늬를 만들었다. 두 결과를 더한 다음 문자가 물 속에서 물결무늬의 영향을 받는 효과를 부각시켰다.

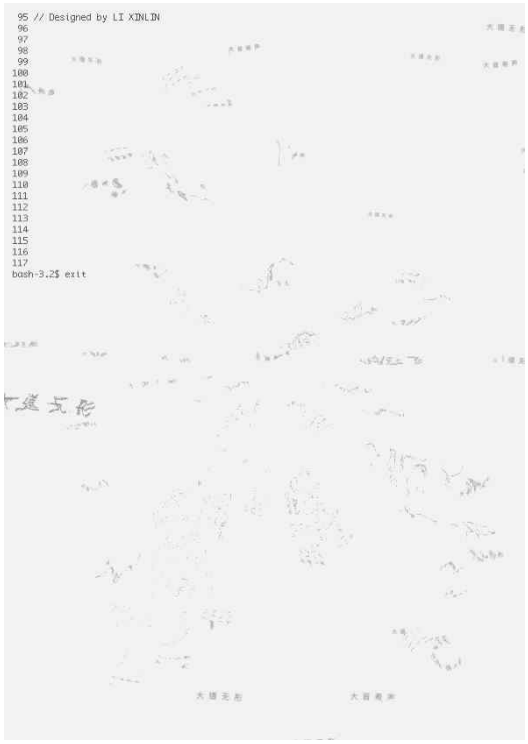
작품 감상 에세이:
<大音希聲，大道无形>
水沒有顏色
但它純洁而又耀眼
水沒有味道
但它清淡而又解渴
水沒有聲音
但它幽靜而又强勢
水沒有形態
但它溫柔而又宏大

<대음희성,대상무형>

물은 색이 없지만
순결하면서도 눈부시다
물은 맛이 없지만
담백하면서도 갈증을 풀어준다
물은 소리가 없지만
고요하면서도 강하다
물은 형태가 없지만
온유면서 방대하다

<The great sound is soundless,
and the great way is shapeless>

Water is colorless,
but it is pure and shining.
Water is tasteless,
But it can quench one's thirst.
Water is soundless,
but it is tranquil and strong.
Water is shapeless,
but it is gentle and powerful.



☑다차원 ☑모듈화 ☑기호화 ☑상형성 ☑ 표의성 ☑은유성

5.4 <弱肉强食> : 약육강식

복합 시스템(Complex systems)의 시뮬레이션에 대해서는 앞에서 이미 논하였다. 생물 개체들이 어떻게 상호 협력하는지를 묘사하여 군집 행위를 묘사하고 피드백 한다. 협력이 있으면 반드시 경쟁도 있게 마련이다. 생물 유전 알고리즘 (Genetic Algorithm) 안에는 생성, 돌연변이, 유전, 진화 등 방법이 있다. 본 작품은 생태 시스템을 시뮬레이션(Ecosystem Simulation) 할 때 생물의 생존과 소멸 방식에 포인트를 두었다. 실재 세계 안에는 각양각색으로 생존하고 발전하는 모습이 존재한다. 약육 강식이 바로 그중 하나다.

본 작품은 생태 시뮬레이션 알고리즘에 생물 진화의 특징인 '사망'과 '탄생'을 더하였고 디자인 요소로 한자를 활용하여 포식자와 대척점에 있는 피 포식자의 행위를 묘사하였다.

작품 감상 에세이:

<弱肉强食>

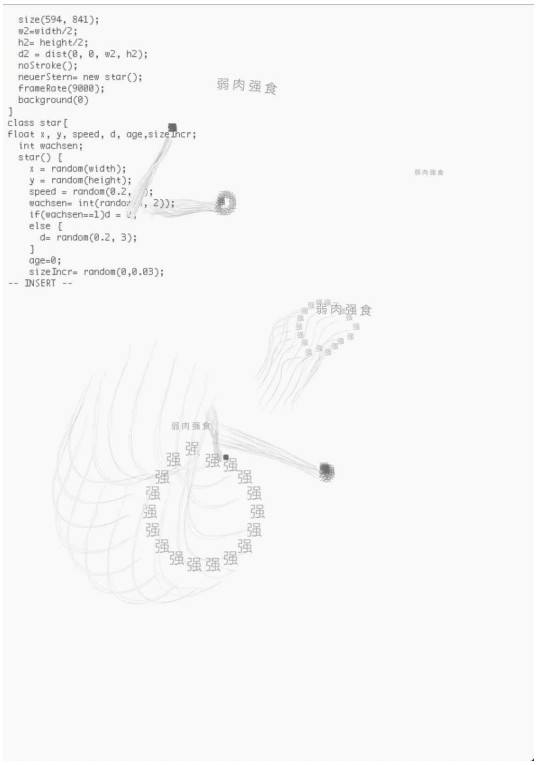
生命既脆弱又顽强
現實世界中的每一个物种
雖然都在弱肉强食下疲于保命
但他們都有相同的生存机會
他們想方設法延續生存時間
等待下一个新的个体誕生

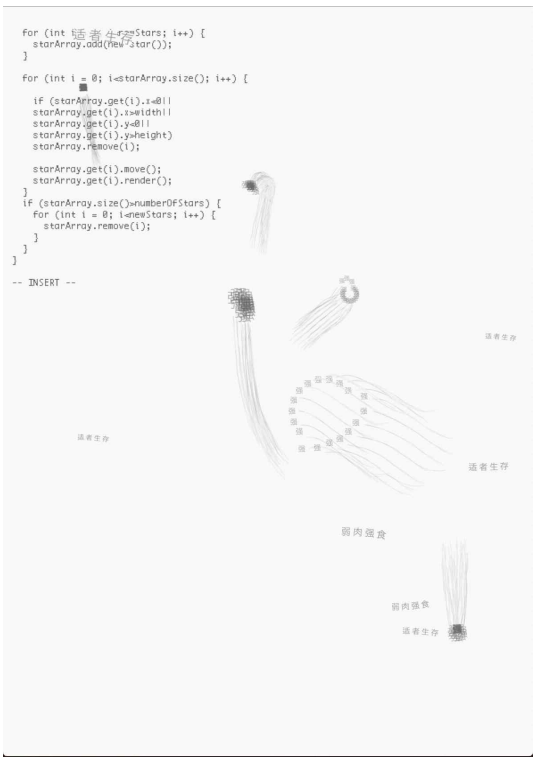
<약육강식>

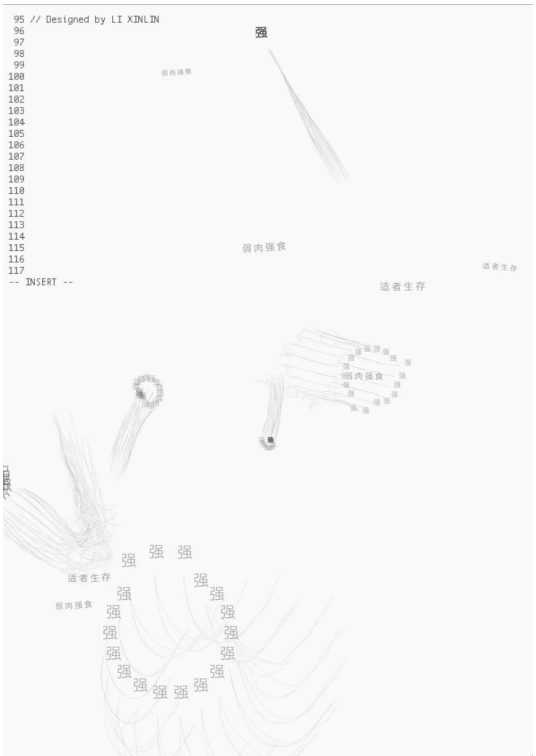
생명은 연약하면서도 강하다
이 땅에 살고 있는 모든 종(種)은
약육강식 아래 목숨을 부지하느라 고단하지만
우리는 모두 똑 같은 생존의 기회를 가지고 있고
온갖 방법을 강구해 생의 시간을 이어가며
새 생명의 탄생을 기다린다

<The Jungle Law>

Life is both weak and strong.
Every creature in the real world struggles to survive
under the jungle law.
Yet they have equal opportunities to live.
They try every way to extend their lives,
waiting for the newly born.







☑다차원 ☑모듈화 ☑기호화 ☑상형성 ☑ 표의성 ☑은유성

5.5 <風回路轉> : 풍회로전

인쇄매체 상의 문자는 정지 상태에 있고 문자의 위치에 따라 독자의 시선이 움직인다. 공간과 시간을 축으로 삼는 폰트는 움직임이 가능하다. 독자의 시선이 줄곧 화면의 한 점에 머무르면 읽는 시간과 순서는 화면에서 움직이는 문자에 의해 결정된다. 매우 순간적이고 선택 불가능한 특성을 지녀 이러한 체험은 눈 깜짝할 사이에 순간적으로 사라진다. 글자의 움직임이 끝나면 인상만 남는다. 이러한 인상은 형제가 없는 무형인데 그 이유는 컴퓨터 모니터, TV 화면이나 영화 스크린과 같은 가상공간 안에 존재하기 때문이다. 작품 창작 시 고려해야 할 사항은 어떻게 하면 폰트가 공간 안에서 더 생동감 있는 이미지로 움직이느냐 하는 것이다. 짧은 시간 안에 관객에게 상상과 함께 또렷한 인상을 남겨야 한다.

우리가 시간의 요소를 공간 안에 넣을 때 표현의 무한성은 더 커진다. 시간이 변화하면서 화면의 공간 역시 계속 전환한다. 하나의 '장소'에서 다른 '장소'로 넘어갈 수 있으며 2차원의 공간에서 하나의 매개체를 선택하면 더 이상 변화하지 못하는 제약에서도 자유롭다. 스크린 공간은 '인터페이스'의 크기가 변화할 수 없는 것을 제외하고, 공간과 공간 안의 요소는 모두 시간의 흐름에 따라 계속 바뀔 수 있다.

작품 <風回路轉 풍회로전>은 고대 한자의 글자 제작법 『육서』의 하나인 '가차'법을 활용하였다. '가차'는 한자의 구성에서 같은 음의 글자를 빌어 딴 뜻에 사용하는 방법을 말한다. 아래 작품에서는 '峰 산봉우리 봉'자 대신 중국어로 동일한 발음의 '風 바람 풍'자를 사용했다.

작품은 사자성어를 차용하여 전방의 구불구불한 길을 표현하는데 상황이 호전될 가능성이 나타난다.

작품의 제작 과정은 다음과 같아. 컴퓨터 안에 여러 원을 계속 만들어내고. Recursion을 사용하여 Circles의 생성 방식을 시뮬레이션 한다. 원의 크기와, 위치, 거리를 설정하고 화면의 원근을 통해 움직이는 공간을 시뮬레이션 한다. 그다음으로 이러한 원들의 가장 자리를 따라 점들을 생성하고, 마지막으로 이러한 점들의 위치를 통해 한자를 전달한다.

작품 감상 에세이:

<風回路轉>

山不轉路轉、
路不轉人轉、
人不轉心轉、
心不轉念轉、
轉心轉念

人生的空間和時間就在寬与窄和長与短中
積窄而寬，積短而長
无所謂憂愁与悲傷
也許就在下一刻出現轉机

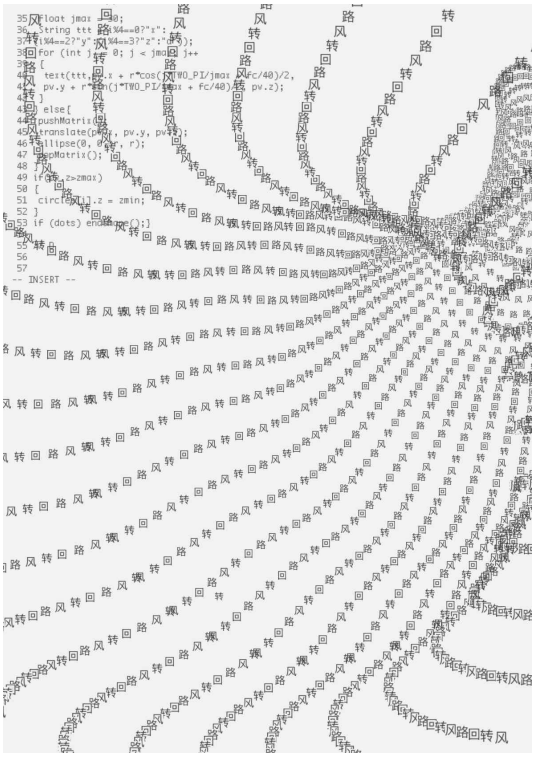
<풍회로전>

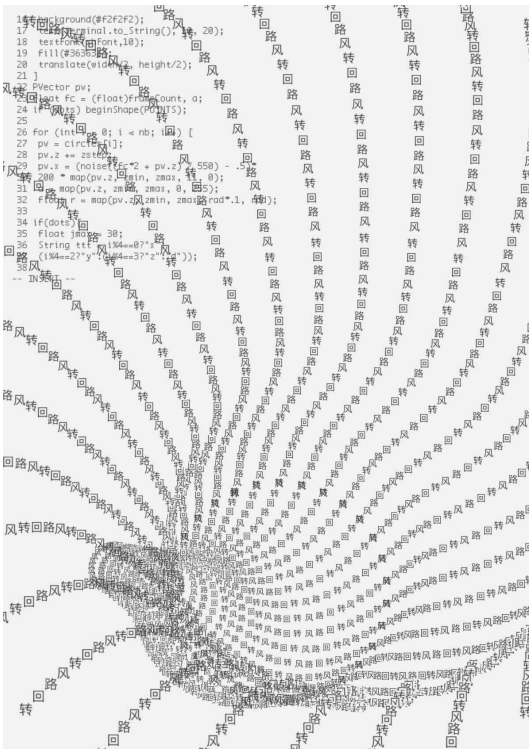
산은 변함이 없고 길이 변한다
길은 변함이 없고 사람이 변한다
사람은 변함이 없고 마음이 변한다
마음이 변함이 없는데 생각이 변한다
마음을 바꾸고 생각을 바꾸자

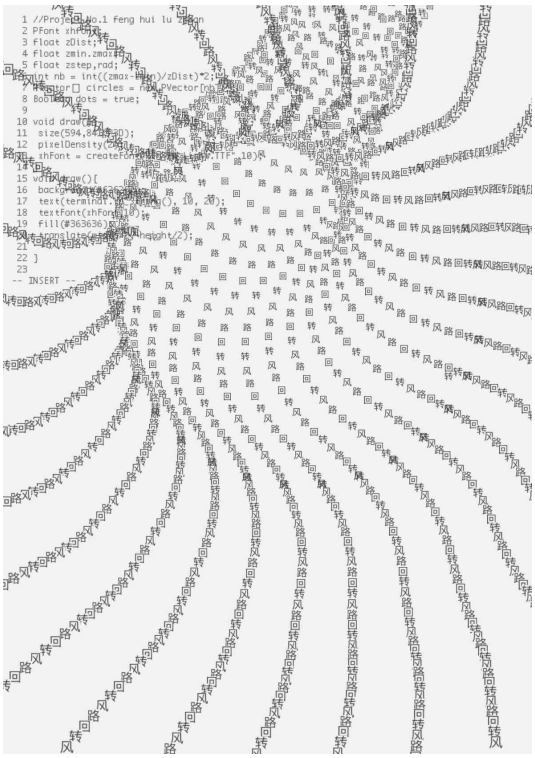
생의 공간과 시간은 넓고도 작고, 길고도 짧은 가운데
좁으면서도 넓고, 짧으면서도 길다
근심과 슬픔을 괘념치 말라
잠시 후 기회가 올지도 모르니 말이다

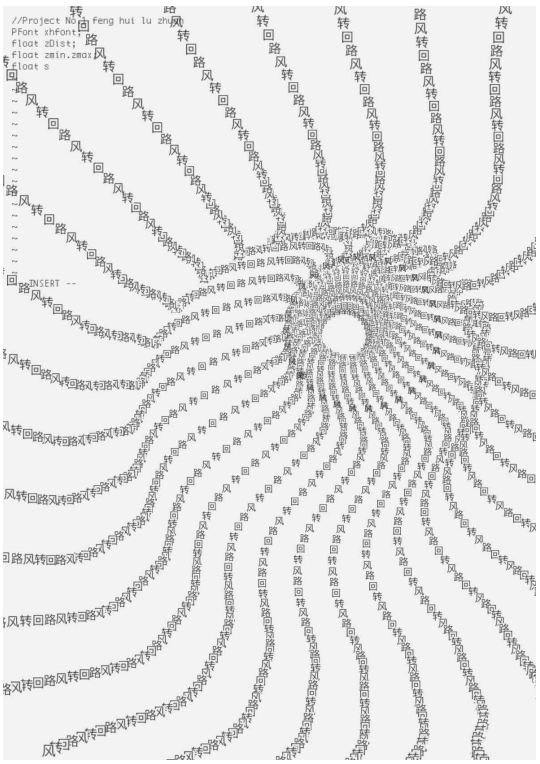
<A Turn of Things>

When the mountain is still, the road can be flexible.
When the road is fixed, one could be flexible.
When one remains the same, his mind could be flexible.
When one's mind is far from changes, his thoughts could be flexible.
Flexible mind, flexible thoughts.
The time and space of one's life lie in t
he width and narrowness, the length and the shortness.
Combining the narrowed into the wide,
accumulating the short into the long.
No need to be trapped in sorrow and blue,
a turn of things may be at the corner.









☑다차원 ☑모듈화 ☑기호화 ☑ 표의성 ☑은유성

5.6 <以大化小 , 以小見大> : 큰 것, 작은 것

앞에서 우리는 전통 키네틱 타이포그래피의 디자인 방법에 따라 한자에 움직이는 배열을 만들고 컴퓨터 시뮬레이션으로 만든 키네틱 모델과 한자를 결합했다.

한자가 다른 문자와 구별되는 또 하나의 중요한 특징은 바로 개별 문자의 구조가 각기 다르고 글자 안에 심오한 의미가 내포되어 있다는 점이다. 모든 문자는 수천 년의 부침과 집약을 거친다. 움직임의 방식을 활용하여 각기 독립되어 있는 한자의 함축적 의미를 어떻게 표현해내는가, 이것이 본 작품의 목적이다.

한자문화권에 속해 있는 아시아 국가는 지리 환경과 민족 성향의 차이로 서로 다른 심미관을 형성하였지만 사실 국가마다 다른 미학 사상은 긴밀하게 서로 연결되어 있다.

중국은 큰 것을 미덕(以大爲美)으로 삼는데 우리는 산수화를 통해 대자연을 바라보는 중국인의 시각에 대해 이해할 수 있다. 중국인은 높은 곳에 올라 멀리 바라보고, 웅장하고 기개롭다고 여기며 그 안에 우주의 원기가 담겨있다는 시각으로 자연을 바라보며 감상한다. 심지어 문자에도 '클 대' 자를 자주 사용하였다. 역대 제왕들은 나라 앞에 '大'자를 붙여"大漢(대한), 大唐(대당), 大宋(대송), 大明(대명), 大淸(대청)"의 표현을 사용하길 원하였다. 또한 문인들 역시 '大'자로 마음속 사물을 칭송하였는데 "大道(대도), 大雄(대웅), 大業(대업), 大音希聲, 大象無形(대음희성 대상무형, 큰 소리는 오히려 들리지 않고, 큰 모양은 오히려 형태가 없다)" 등이 바로 그 예이다. '클 대'와 반대되는 "작을 소"자 안에도 동양 미학 사상이 담겨 있다. 그 안에는 간소화, 디테일 중시, 섬세하면서도 꼼꼼하게 대상을 주의 깊게 바라보는 가치관이 존재한다. 이것은 비단 중국만의 사고 방식은 아닐 것이다. '大'와 '小'의 철학은 동양인의 피 안에 흐르고 있다. 이에 본 작품에서는 '大'와 '小' 두 글자를 사용해 창작을 진행하였다.

작품에는 기술적으로 복잡한 알고리즘을 사용하지 않고 Processing이

제공하는 기본 함수 예를 들면 그래픽 처리, 화소 판독, 클로즈업 등의 기능을 사용하였다.

작품 감상 에세이:

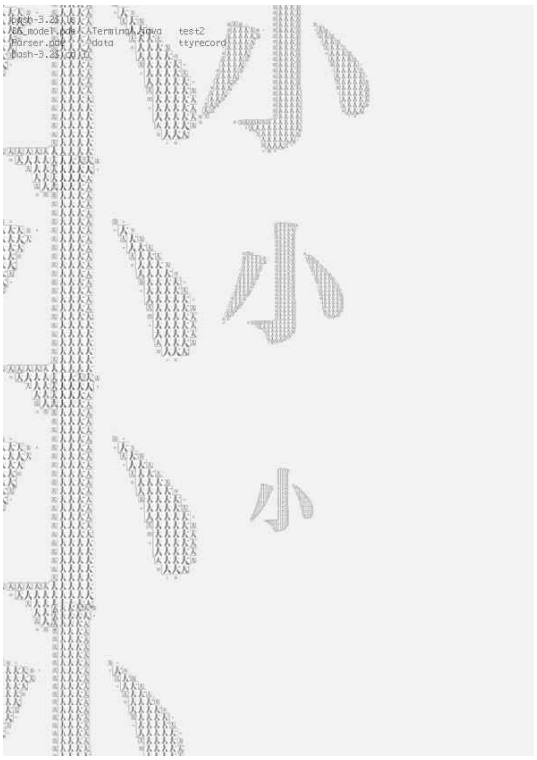
<以大化小，以小見大>
從局部見整体
從整体見局部
從有限見无限
從无限見有限

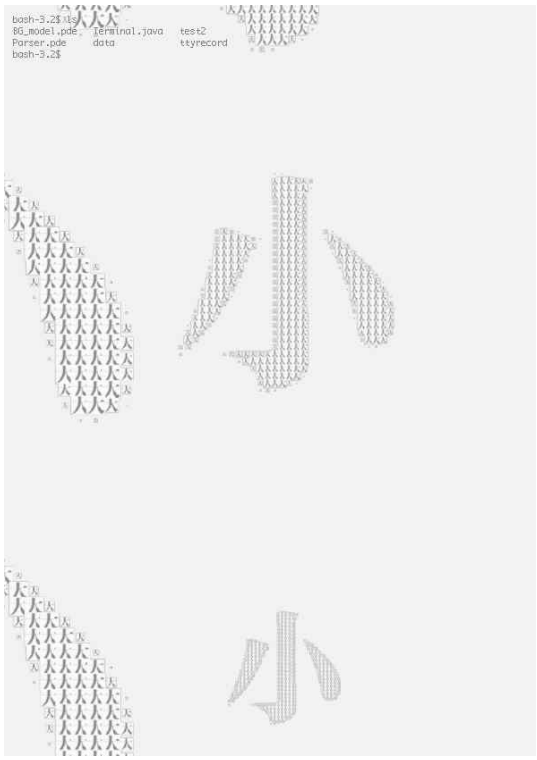
<以大化小 以小見大>
: 큰 것을 작은 것으로 만들고, 작은 것으로 크게 바라보네

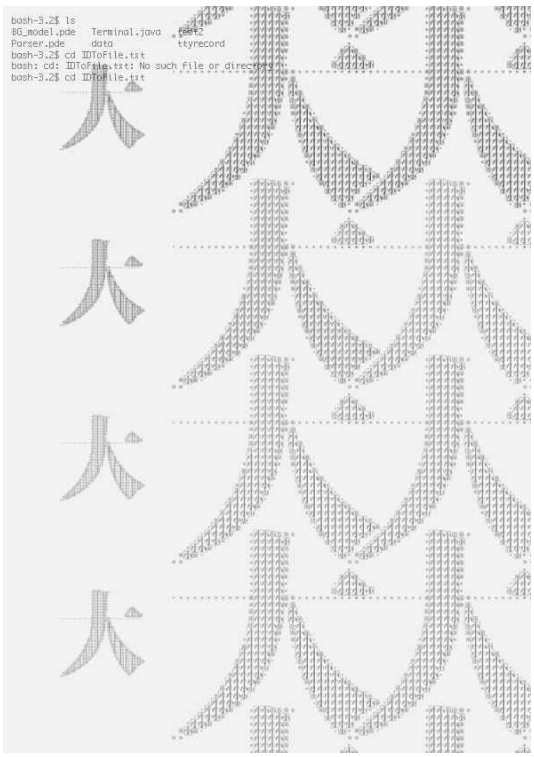
일부로 전체를 바라보고
전체로 일부를 바라보며
유한으로 무한을 바라보고
무한으로 유한을 바라본다

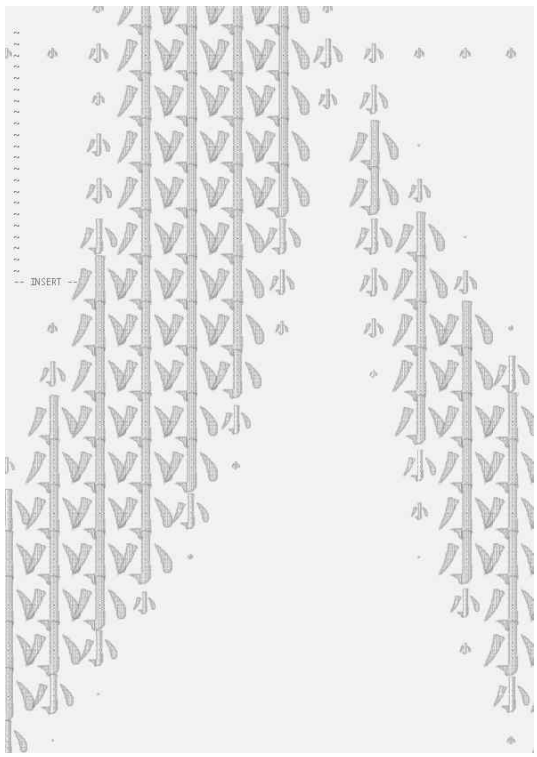
<Understanding Each Part from the Whole
and the Whole from a Part>

From a part, we could see the whole
From the whole, a part could also be seen
From the limited, we could understand infinity
From the infinity, the limited could be understood as well









☑다차원 ☑복합성 ☑모듈화 ☑형태의 시각적 시점 ☑기호화 ☑ 표의성 ☑은유성

5.7 <你中有我，我中有你> : 너 , 나

시리즈 작품 <你中有我，我中有你>은 5.6 작품 <대소>와 대조를 이루며 움직이는 방향도 상반되게 표현했다.

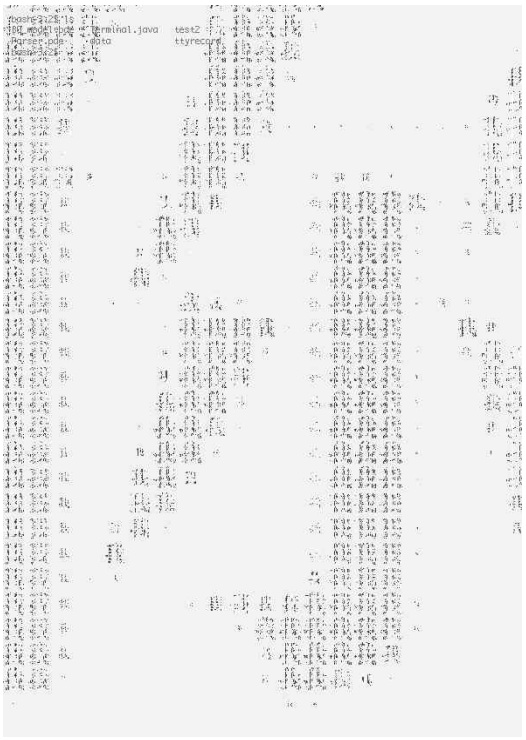
작품 감상 에세이:

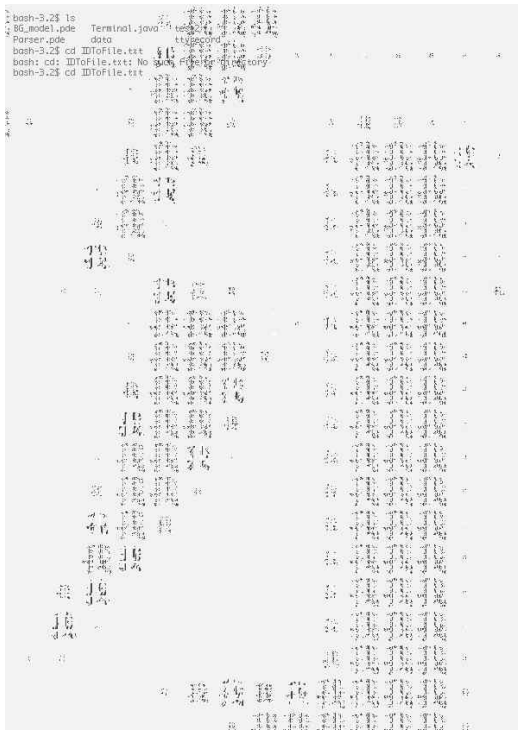
<你中有我，我中有你>
当兩人之間不斷産生聯系
就可以无視速度的存在
共享同一个時間
共享同一个空間
共享人間

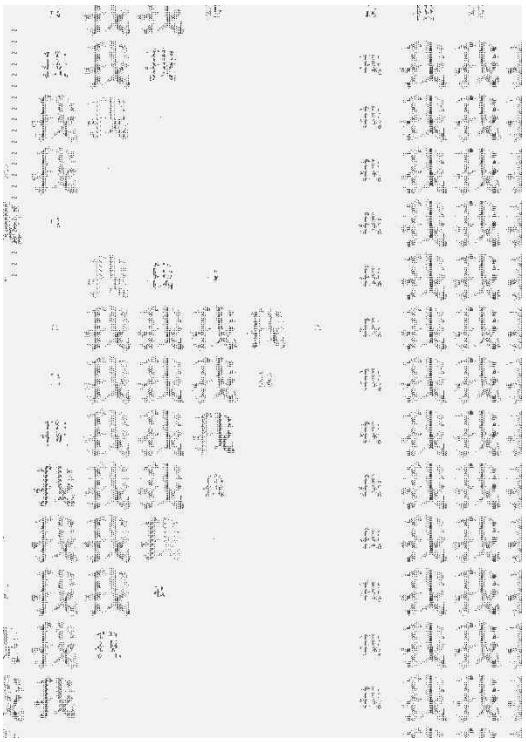
<네 안에 내가 있고, 내 안에 네가 있다>

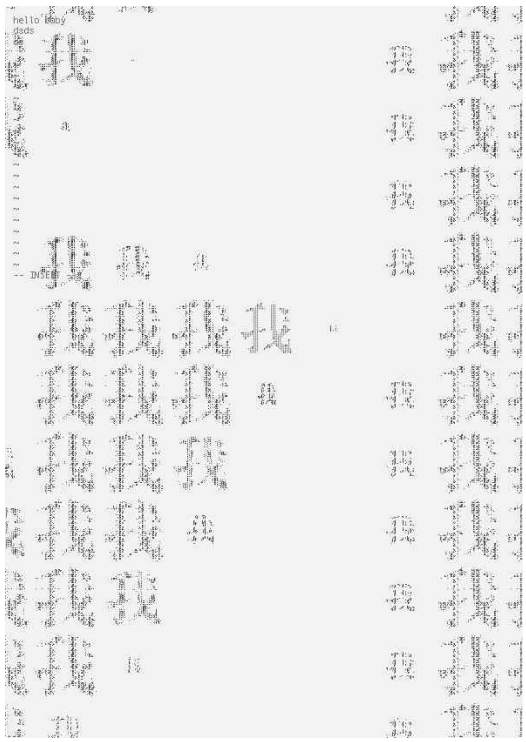
두 사람은 계속 연결되어 있어
속도의 존재를 무시할 수 있다
같은 시간을 나누고
같은 공간을 나누며
세상을 함께 누린다

<You, Part of Me and Me, Part of You>
When two are interconnected all the way,
we can ignore the very existence of speed.
We, share time, space, and the world.









☑다차원 ☑복합성 ☑모듈화 ☑형태의 시각적 시점 ☑기호화 ☑ 표의성 ☑은유성

5.8 <言多必失> : 말은 실수를 낳는다

한자 서체의 변환 과정은 3단계로 나누어 볼 수 있다. 대전(大篆)에서 소전(小篆)으로 소전에서 예서(隸書)로, 예서(隸書)에서 다시 해서(楷書)로 변하였다. 해서가 등장한 이후에도 일상에서 사용하는 행서(行書)와 초서(草書)가 또 등장한다. 오늘날 사용하는 한자와 비록 서체 모습이 다르지만 한자의 기본 구조와 쓰는 방법은 이미 고정되어 있다. 앞에서 우리는 한자의 "표의성"에 대해 논하였다. 개별 한자의 표의적 특징을 한 층 더 두드러지게 표현하기 위해 본 작품을 창작하였다.

인간은 '칠정육욕'이 있는데 '말하고 싶은 욕구'도 그 안에 포함된다. 우리는 종종 말하고 나서 그 때문에 후회한다. 이에 본 연구자는 인간은 너무 많은 말을 해서 안되며 말을 삼가고 조심해야 한다는 내용을 경고하기 위해 작품을 창작하게 되었다. 작품의 제목도 말이 많으면 반드시 실수한다는 중국 사자성어 <言多必失>로 삼았다. 에세이에는 헤밍웨이가 쓴 "Two years to learn to speak, a lifetime to learn to shut up." 구절을 인용하였다.

작품 창작 시에는 한자 "言" 말씀 언 자의 글꼴 특징을 두드러지게 하고 글자 하단 부에 "口" 입 구 자를 남겨두었다. 상반 부분에는 무한하게 증가하는 "橫" 가로 횡 필획을 사용하여 계속 언어를 말하는 모습을 표현했다. 다양한 사람과 다양한 방언, 다양한 내용을 말하는 모습을 비유하고 싶은 목적에서 "橫"자는 다양한 폰트를 사용하였다.

본 작품은 입자 시스템(Particle Systems)의 논리를 사용하여 입자가 위치를 바꾸고 사라지는 방식을 시뮬레이션 하였고 사진으로 입자 하나하나의 위치를 로딩하였다.

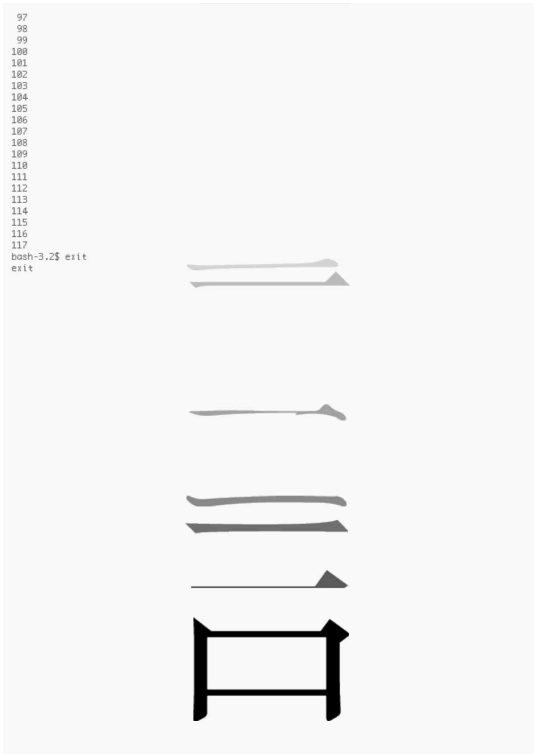
작품 감상 에세이:

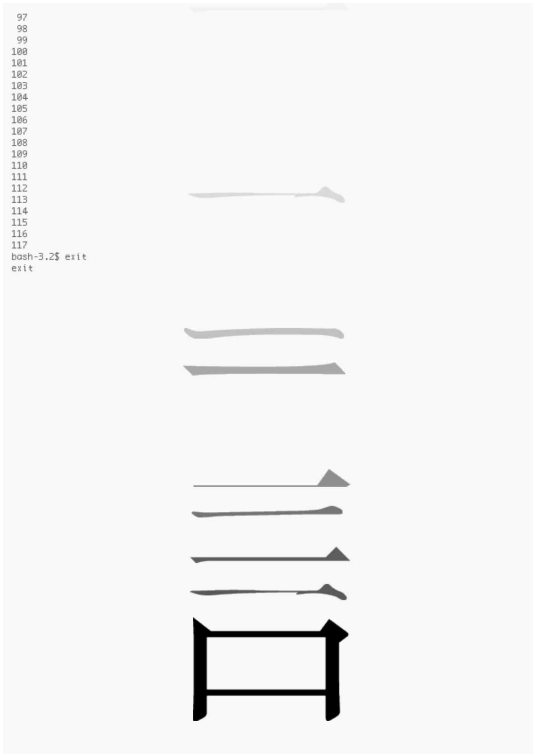
<言多必失>　　<말은 실수를 낳는다>

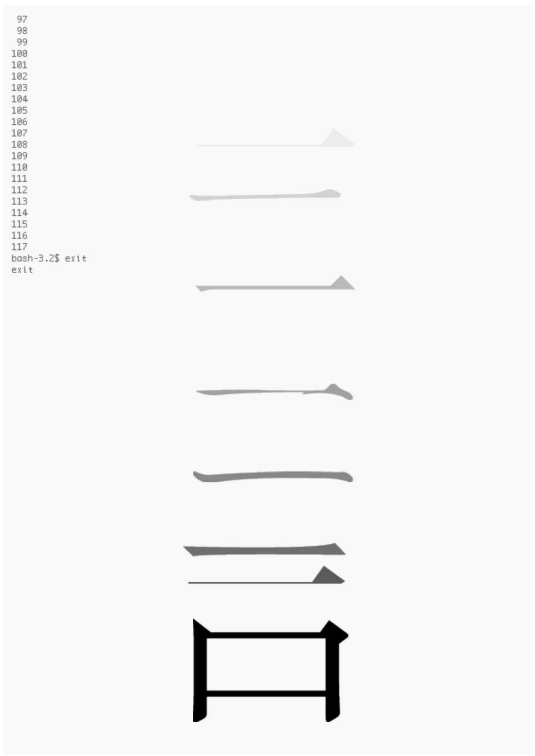
我們花了兩年學會說話　　우린 말을 배우는 데 2년이 걸렸지만
却要花一輩子來學會閉嘴　　침묵을 배우는 데는 평생이 걸린다
大多數時候，我們說的越多　　많은 경우 말이 많을수록
彼此的距离越遠　　서로의 거리는 점점 멀어지고
矛盾也越多　　갈등도 점점 많아진다
我們急于表達自己　　자신을 표현하기 위해 급급할 뿐
却一点不懂對方感受　　상대의 감정에 대해서는 전혀 알지 못한다

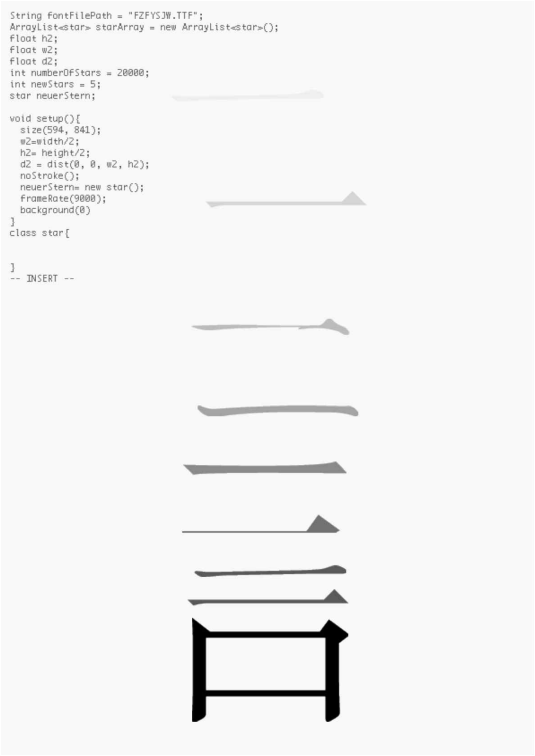
<More Words, More Pain>

Having spent two years learning to speak,
but we have to learn another lesson -- shutting up, for our lifetime.
Mostly, the more words we say,
the more distance and contradiction grow between us.
Sometimes, we are eager to express what we think
without understanding of what others feel









☑다차원 ☑복합성 ☑모듈화 ☑형태의시각적 시점 ☑기호화 ☑ 표의성 ☑상형성 ☑은유성

5.9 <合二爲一> : 둘이 모여 하나가 되다

디자인 할 때 우리는 다른 뜻을 내포하고 있는 한자를 배열하고 조합하여 공통의 한자를 표현할 수 있다. '한자를 조합하여 나온 형태(방식)와 그 의미를 어떻게 일치시키느냐' 이 문제는 본 작품의 주요 연구 대상이다.

작품의 주제로 철학사상 "合二爲一"을 택한 이유는 아래와 같다.

첫째, 중국 전통문화는 줄곧 유교와 불교, 도교 3대 종교의 영향을 받았다. 유교 학설의 '천인합일'과 불교의 두 손을 모으는 합장 그리고 도교 태극의 음양흑백 안에는 모두 하나를 나누면 둘이 되고, 둘을 합하면 하나가 된다는 '一分爲二, 合二爲一' 사상이 담겨있다.

둘째, 문자 구조 형태 측면에서 바라보면 '二'은 두 개의 '一'을 조합하여 만든 것인데 한자는 내포하는 의미와 그 형태가 서로 연관되어 있다.

작품을 제작할 때에는 먼저 일정 수량의 입자 (Particle) 을 생성하고 나서 입자의 생산, 집합, 경계판정 및 사라짐을 제어하였다. 마지막으로 입자 위치를 한자 하나하나에 다 전송하였다.

작품 감상 에세이:

<合二爲一>

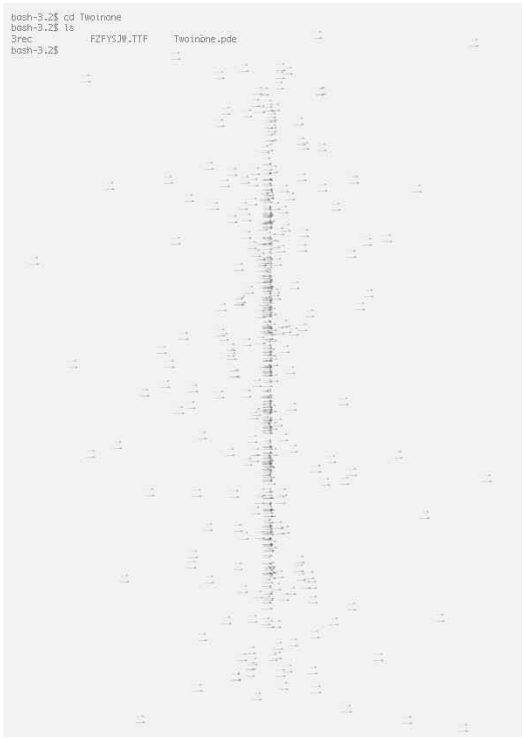
放弃成爲个体
才有了整体
雖不分左右
却卽左也右
道法自然
万物歸一

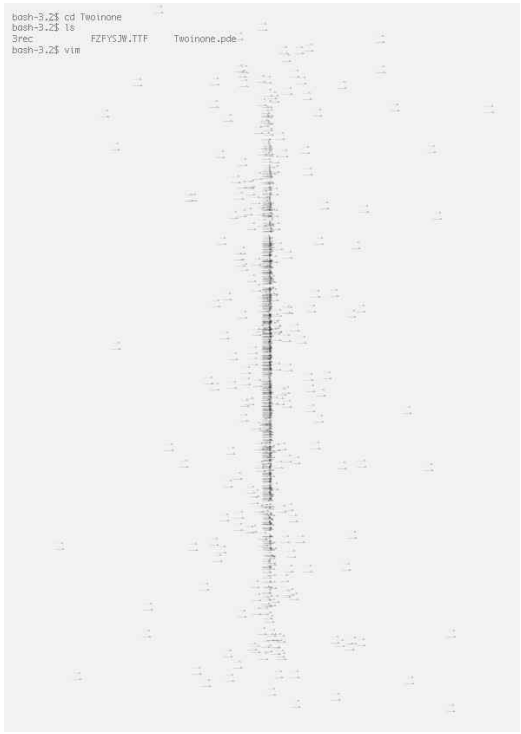
< 둘이 모여 하나가 되다>

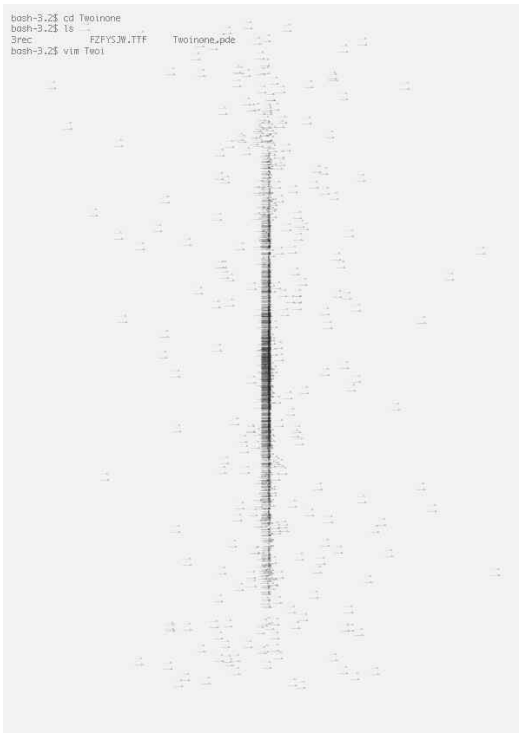
개인이길 포기해야
전체가 된다
좌우를 나누지 않아도
좌우는 있다
법칙은 자연스러워
만물은 모두 원점으로 돌아간다

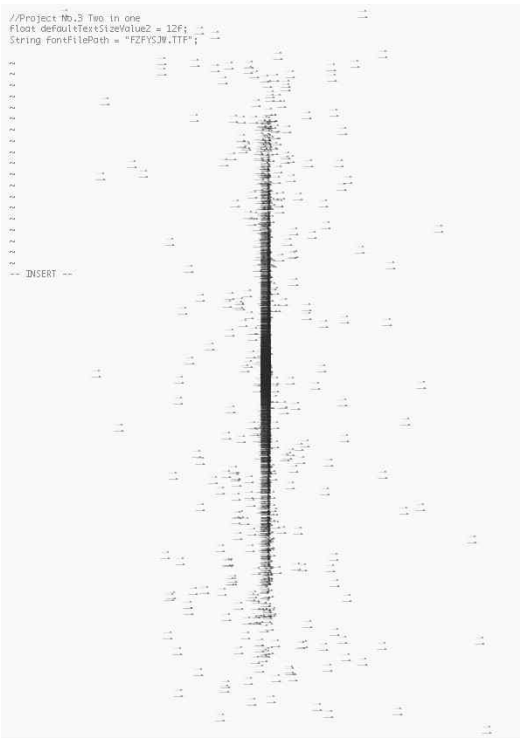
<Combining as One>

Giving up the isolated state,
one becomes the whole.
As the left is not distinguished from the right,
it could both right and left.
The way from what is beneath abstraction,
everything combines as one.









☑다차원 ☑복합성 ☑모듈화 ☑형태의시각적 시점 ☑기호화 ☑ 표의성 ☑상형성 ☑은유성

5.10 <一 木> : 나무 한 그루

동양의 많은 예술가는 자연 모방을 추구한다. 하지만 동양에서 자연을 모방하는 방식은 서양의 리얼리즘 예술과는 다르다. 동양 예술가는 완벽한 동일을 추구하지 않으면서 자연의 법칙을 파악하고 싶어 한다.

자연이 만물을 창조하는 원칙은 한 그루의 나무와 같다. 나무의 줄기와 가지, 나뭇잎을 자세히 바라보면 서로 흡사해 보이지만 이들이 조합하여 자란 나무는 각기 다르다. 개별 한자도 한 장의 나뭇잎과 같아 비슷해 보이는 재료를 사용하여 각기 다른 텍스트를 만들어 낸다.

“木” 자는 나무를 가리킨다. 여기에 ‘一 ’자를 더하면 ‘本’근본 본 자가 되는데 “本”은 나무의 뿌리 부위를 가리킨다. 뿌리는 나무의 기원이기에 작품에서는 한자 “本” 중의 “一”자를 사용하여 나무뿌리를 대신하였다. 동시에 한자의 근본은 “필획”에 있음을 표현하고자 했다.

작품 <一木>은 Fractals 개념을 도입하여 Recursion을 사용해 Trees의 성장 방식을 시뮬레이션 한다. transformation matrix를 사용하여 나뭇가지의 생장 각도를 계산했다.

작품 감상 에세이:

< 一 木>

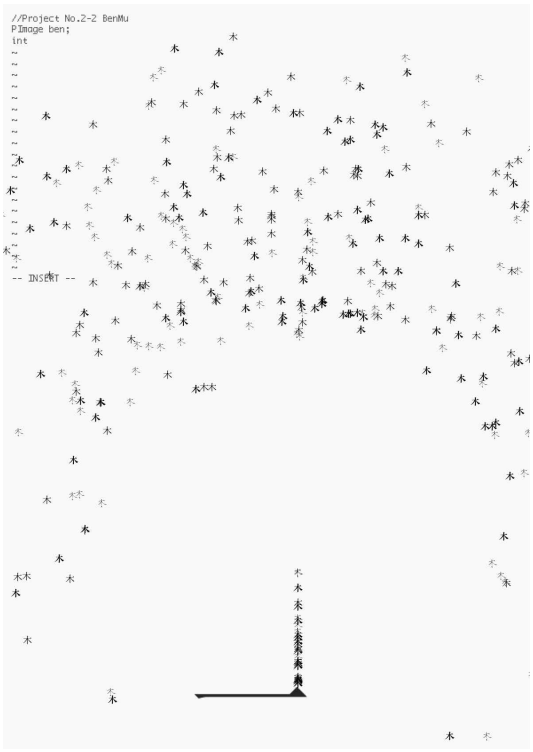
眞實的自然中,
一切其實都是不能預測的。
瞬息万變的世界里,
變化是每一刻日常生活中的常態,
一切都是在連續變化的狀態上成立的。
生命的无常,
讓我們更了解整个世界。

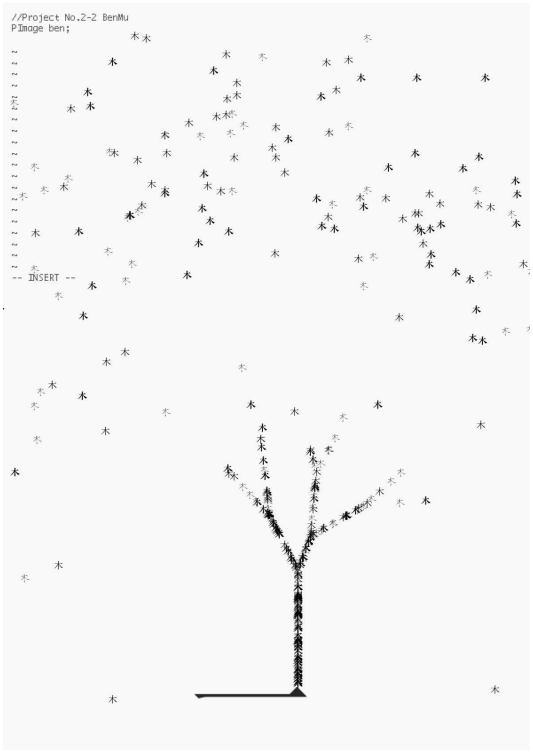
<나무 한 그루>

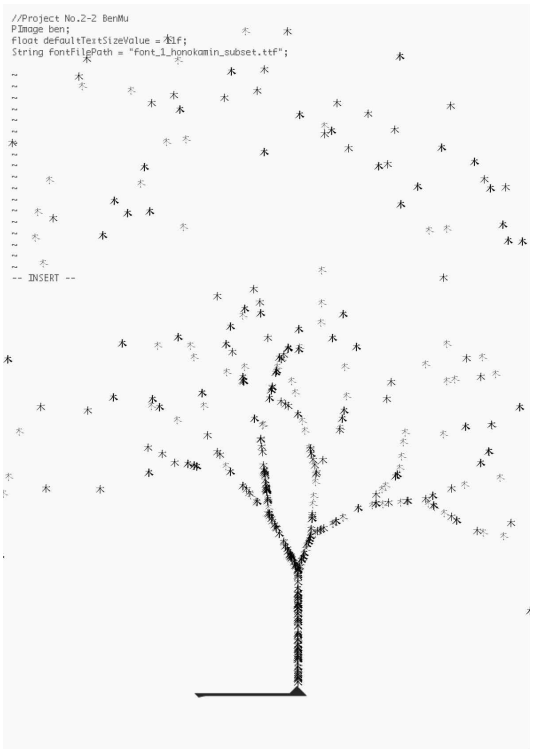
실재하는 자연에서
예측할 수 있는 것은 없다
순간순간 변하는 세상 속에서
변화는 매 순간의 일상이며
모든 것은 계속 변화하는 상태에서 만들어진다
생명은 변화무상하여
우리에게 세상을 더 잘 이해하게 만든다

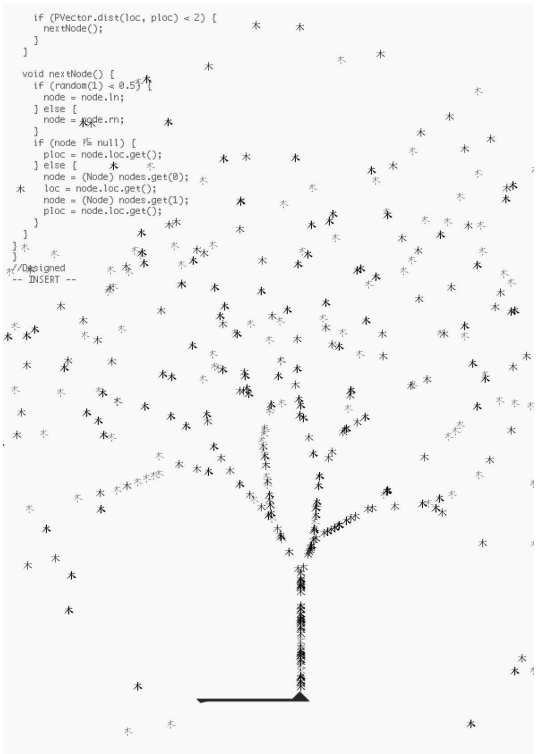
<Wood>

In the great nature,
everything is indeed unpredictable.
In this fast-changing world,
What does not change is changing itself.
Everything is based on the state of being changing continuously.
And the impermanence of life enables us to understand this world better.









☑다차원 ☑모듈화 ☑형태의시각적 시점 ☑기호화 ☑ 표의성 ☑상형성 ☑은유성

5.11 <一 草> : 풀 한 포기

“無常”무상은 불교와 선학(禪學)의 가장 기본적인 개념으로 중요 사상의 내용 중 하나다. 무상은 세계(시간과 공간상)의 모든 물체를 말한다. 감정이 있든 없든 형태가 있든 없든 모두 만들어지고 없어지면서 변화를 이어가고, 부단히 재구성하고 바뀌면서 새로 만들어지거나 사라진다. 사실“무상”은 매 순간 부단히 발생하고 있는 일상 현상이다. 작품 <一 草>는 이전 작품들의 스타일을 이어가는 동시에 무상의 미(美)를 전달하고 싶은 취지에서 창작하였다.

본 작품은 <一 草>를 주제로 풀의 형태를 추상화하였고 한자 “草”자를 분해하는 방식으로 경지를 표현했다. “草”는 “艹”풀 초자와 “早”아침 조 자가 결합되어 형성된 글자다. “艹”자는 한자에서 초본식물을 상징한다. 작품에서는 “草”자의 특징과 풀뿌리 식물의 외부 형태를 결합했다.

본 작품은 Fractals 개념을 도입하여 Recursion을 사용해 Trees의 성장 방식을 시뮬레이션 했다. transformation matrix 사용 시 임의 수를 더해 나뭇가지의 생장 각도를 계산했다.

작품 감상 에세이:

<一 草>

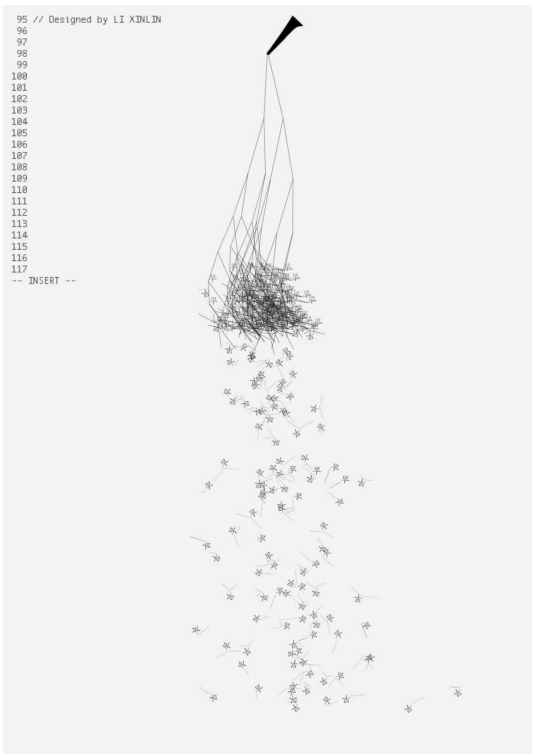
春去冬來，
寒暑交替，
日月消長，
万物生滅，
生命一切，
沒有永恒，
只有无常。

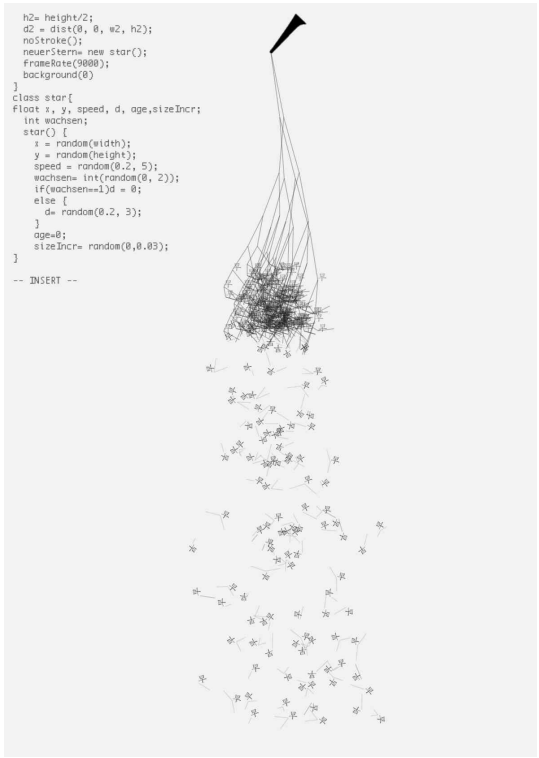
<풀 한 포기>

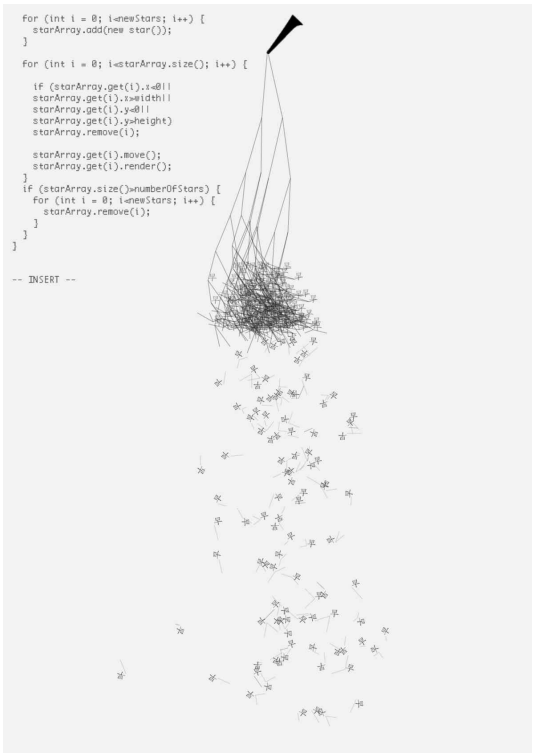
봄이 가고 겨울이 오면
추위와 더위가 교체한다
날과 달이 차고 이지러지면
만물은 태어나고 소멸한다
생명은 모두
영원하지 못하고
무상(無常) 할 뿐이다

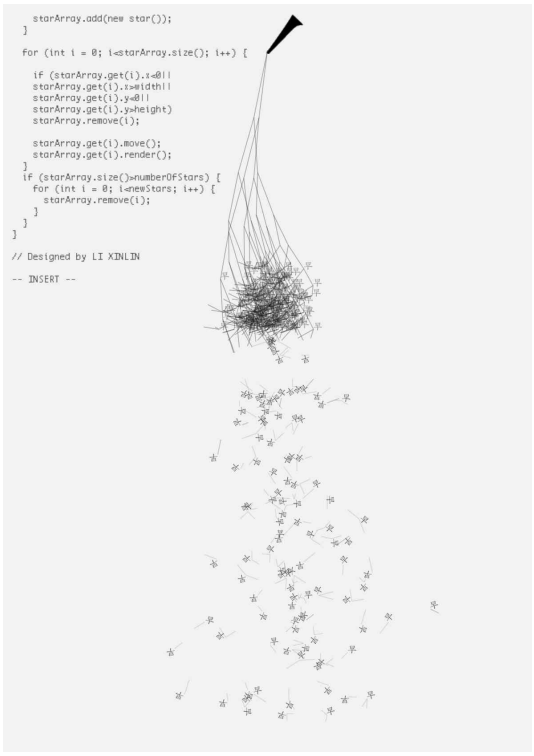
<Grass>

From spring to fall,
From summer to winter,
The sun and the moon are changing forever.
The creatures in the world are growing and dying.
There is no eternity,
but impermanence never fades away.









☑다차원 ☑복합성 ☑모듈화 ☑형태의시각적 시점 ☑기호화 ☑ 표의성 ☑상형성 ☑은유성

5.12 <一 花> : 꽃 한 떨기

일본 디자이너 우치다 시게루는 『전후 일본 디자인 역사』에서 21세기 디자인의 존재 방식을 묘사했다. 그는'Weak Moderlity (연약한 감각의 현대성)'이라는 미래적 개념을 남겼다. 미학에서 '연약한 감각'의 디자인은 미묘한 차이를 낳는다. 만지면 부서질 것만 같은 가냘픈 것, 작고 여리면서 보드랍고 매끄러운 것 등등이 바로 그 예이다. 동양은 자연스럽게 이러한 '연약한 감각'의 특질을 갖는다.

시리즈 작품 <一花>은 이러한 '연약한 감각'의 개념을 인용하여 꽃의 형태를 추상적으로 표현했다. 그리고 "花"자를 분해하여 부분적으로 한자의 구조를 남겨두었다. 꽃이 흔들거리는 효과는 2D transformation의 rotate 등의 함수를 사용했다.

작품 감상 에세이:

<一 花>

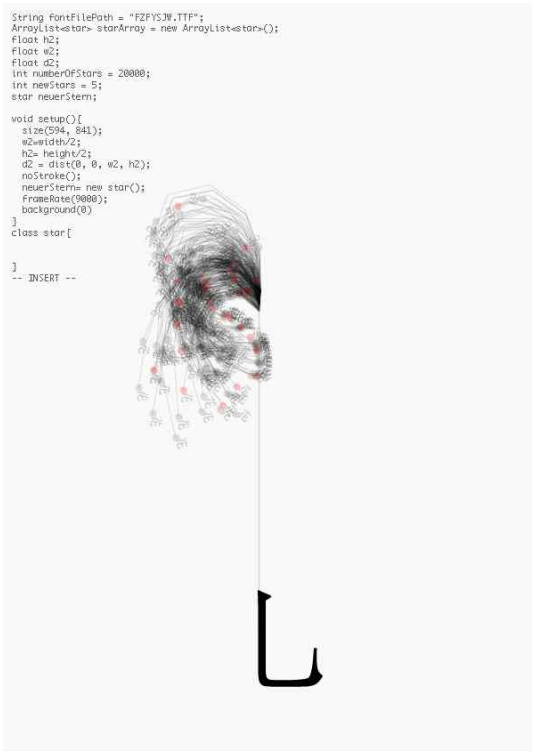
花是柔而不定形的
它具有一种纖細和微弱的狀態,
其中沒有傲慢,
沒有凶暴,
而是溫柔与善意,
寂寞与寂靜,
是內心所感受到的渺茫与易變。

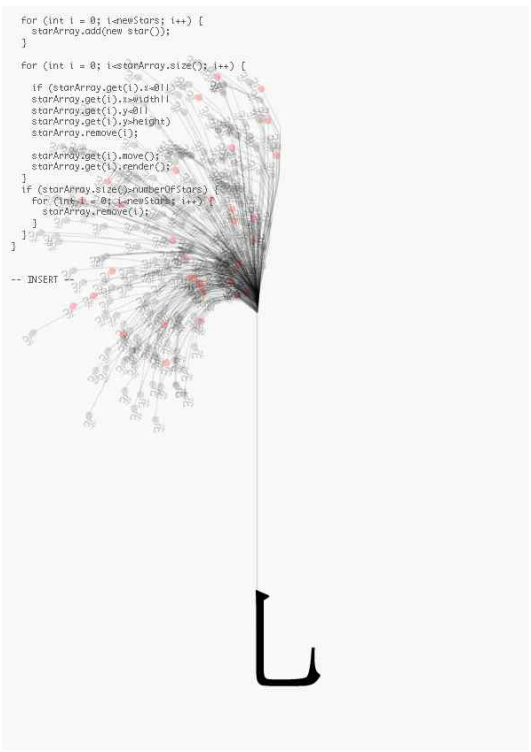
< 꽃 한 떨기>

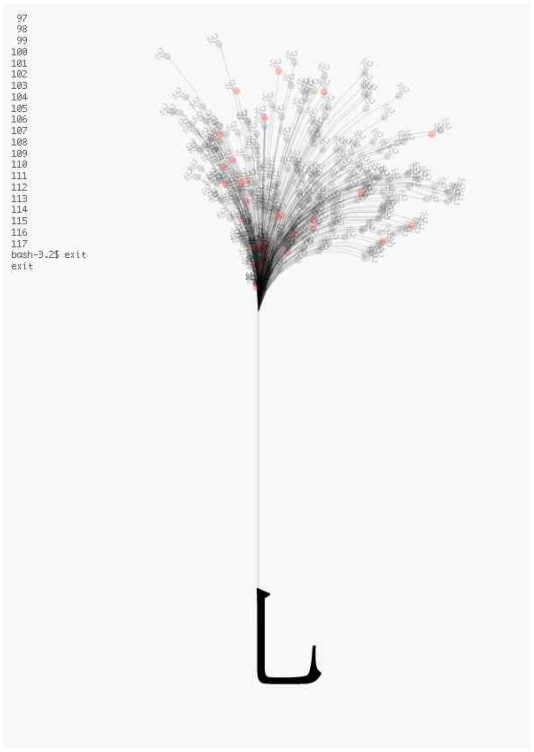
꽃은 부드러우면서도 정형적이지 않다
미미하고 약한 가녀린 상태에서도
오만함이 없고
사나움이 없으며
부드러움과 선함이 있다
외로움과 적막은
내면에서 느끼는 아득함과 불안정이다

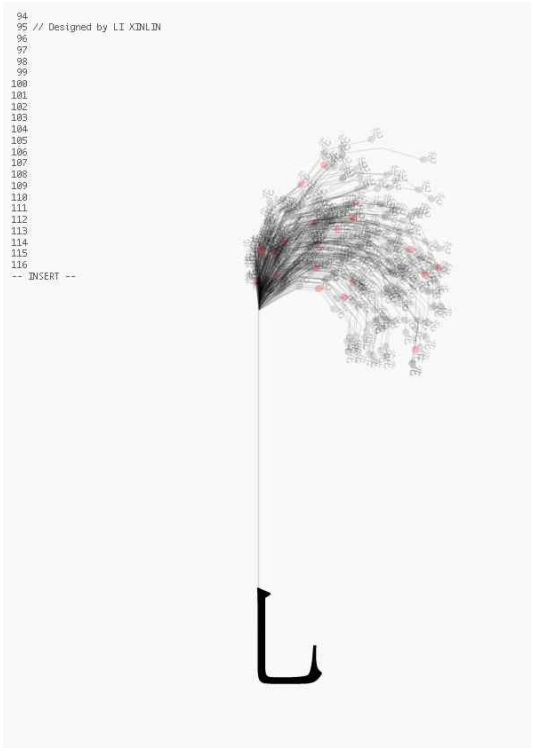
<A Wild Flower>

The gentle and shapeless flower,
in a delicate and weak state.
It has no arrogance,
no violence,
but gentleness and kindness,
but loneliness and tranquility,
the vastness and changeability that you feel in your heart.









☑다차원 ☑복합성 ☑모듈화 ☑형태의시각적 시점 ☑기호화 ☑ 표의성 ☑상형성 ☑은유성

키네틱 한자의 정수 “선(線)”

한자를 전달하는 “매개체”가 계속 바뀌면서 한자의 서체 또한 이에 따라 변화하였다. 한자 서체는 거북이 등껍질 위에 새긴 갑골문(甲骨文), 청동에 새겨 넣은 금문(金文), 비단에 쓴 백서(帛書), 종이 위의 인쇄체, 디지털 매체에 사용하는 점행렬식 Dot-matrix-fonts와 점글꼴 Bitmap-fonts 등의 변천 과정을 거쳤다. 몇천 년의 시간을 거듭하며 “현대 한자”는 “고대 한자”의 서체와 큰 차이를 보이지만, 고대 한자의 정수와 그 안에 내재되어 있는 심미적 아름다움은 여전히 계속 남아있다.

서체별 디자인 스타일을 살펴보면 한자 서체는 대략 “대전, 소전, 예서, 해서”의 4 단계를 거쳤는데 “해서체” 단계에서 전체적인 한자 체계가 완성되었고 사용 범위도 고정되어 바뀌지 않았다. 서법(書法) 중에서도 “해서(楷書)”는 광범위하게 사용되었던 공식 표준 서체로서 서체 중에서 가장 단정하고 곧은 형태의 규범적 필획을 보이는데, 사각 형태 안에 글자 하나하나의 획을 바르게 쓰는 것이 중요하다. 이와 상반되는 서체로는 민간에서 주로 일상생활에 사용했던 “초서(草書)”와 “행서(行書)”가 있다. 이 서체들은 엄격하게 지켜야 하는 규칙이 없었으며 서법(書法)에서도 가장 “본질”적인 서체이다.

초서(草書)의 짜임새는 다른 한자 서체와 달라 쓰는 법이 달랐다. 시간을 아끼고 수고로움을 덜기 위해 빠르게 쓴 까닭으로 다소 거친 형태를 보인다. 초서체는 필획이 매우 간단하고 쓰는 방식도 매우 개성적이다. 사실 한자 문화권에서 생활하는 사람들은 이러한 서체 방식에 매우 익숙하고 글자 사이 행간의 미묘한 차이에 대해서도 잘 알고 느낄 수 있다. 초서체는 서예가의 손을 거치면 한 폭의 서예 예술로 변모하는데(그림 44), 이때에는 문자의 식별 정도보다는 창작자의 직관을 표현하고 내면적으로 느끼는 바를 함께 나누는 것이 더 중요하다.

초서(草書)의 주요 특징은 다양한 질감을 표현하는 “필선”에 있다. 다양한 재질과 다양한 서예 도구의 사용 심지어 개개인이 표현하고자 하는 차이에 따라 서로 다른 결과물이 주어지지만, “선(線)”에 대해 공통적으

로 갖고 있는 심미적 아름다움은 항시 바뀌지 않는다. 선(線)은 그 자체가 추상적, 함축적이어서 감상자에게 구애됨이 없고 자유롭게 느낄 수 있는 여백의 공간을 가져다주는데 이는 동양인의 성향에 매우 부합한다. 대만 디자이너 허자싱(Ho Chia-Hsing)은 그래픽 디자인 아시아 2019 비전 포럼에서 동양 예술의 정수는 “필선”이라고 언급했다(그림 45).

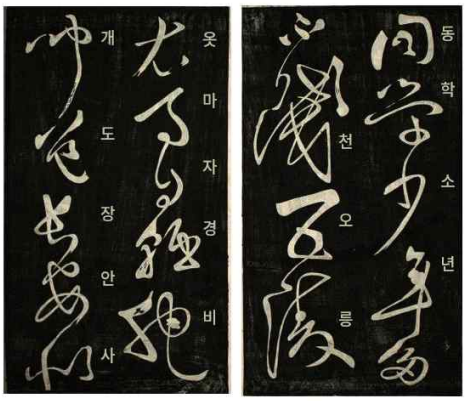


Figure 44. 회소 초서 <추흥팔수> 31)

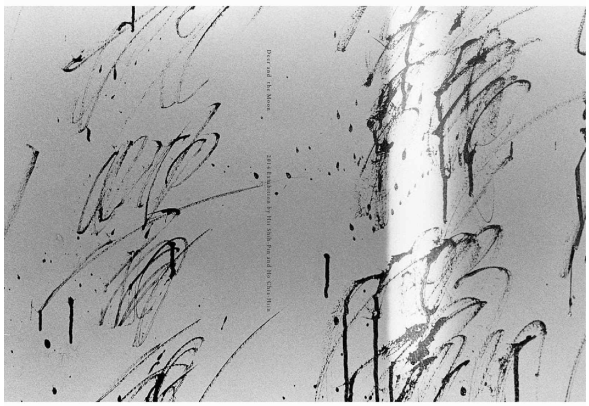


Figure 45. 대만 허자싱 작품 32)

한자 필선(線)의 아름다움은 아래의 몇 가지로 구체적으로 나누어볼 수 있다.

구조

초서의 서체 구조에서 본래 복잡했던 한자 필획은 모두 간략하게 바뀌고 식별 가능한 가장 기본적인 윤곽만 남게 되었다. 간소화된 방법은 구체적으로 필획을 이어 쓰는 “연필(連筆)”법과 필획을 삭제하거나 원래 구조 위에 새로운 형태 등을 추상적으로 그려내는 것으로 나누어 볼 수 있다. 그 밖에도 자간과 행간이 좁고 넓은 정도, 여백 남기기, 쓸 부분만을 헤아리지 않고 쓰이지 않는 부분까지도 고려하는 “계백당흑(計白當黑)” 등의 전통적인 방법을 사용하는데 사실상 선으로 공간을 분할하는 것이다.

31) 懷素,秋興八首,매일뉴스 https://kknews.cc/culture/b4jlp9.html
32) 何佳興 (Ho ChiaHsing) http://studio-bin.blogspot.com/2014/04/41458.html

필법(筆法)

필법에서 강조하는 바는 붓을 위로 일으키거나 아래로 엎는 상하 움직임을 활용하는 것이다. 들어 올리고, 누르고, 붓을 잠깐 멈추고, 아래로 내리고, 방향을 틀고, 꺾는 운필(運筆) 방식을 사용해 서법에서의 필선(線)의 힘과 속도 그리고 부드러우면서도 강인한 정도를 표현한다. 선(線) “힘(力)”의 강약은 직접적으로 문자의 예술성 표현과 생동감에 영향을 미친다. 운필(運筆)의 급함과 완만함, 가볍고 무거운 정도로 다양한 질감의 선(線)을 만들어 낼 수 있다.

묵법(墨法)

먹과 종이는 각기 다른 재료의 질감으로 시각적인 느낌을 다양하게 만들어낸다. 진한 먹물은 화선지 위에 스며들어 선의 윤곽을 두껍고 진하게 만들고, 옅은 먹물은 종이 위에서는 맑고 옅은 선을 만들어낸다. 동양에서는 묵을 검은색 하나로만 바라보지 않는다. 묵을 흑색(焦), 짙음(濃), 회색 (灰) , 옅음 (淡) , 맑은(淸) 다섯 가지 색으로 나누어 농담을 달리해서 [33] 필선의 색을 풍부하게 만든다.

리듬감

동양 회화에서 선은 곧 리듬을 말하는 것이기도 하다. 첩첩이 이어지는 산의 굴곡, 물이 흐르는 시내, 지붕 처마의 높낮이 등이 바로 그 예이다. “한자”는 회화 예술을 직접 표현하는 문자로서 그 안에 담겨있는 선은 입체감과 율동감을 가진다.

한자를 쓰는 과정은 역동적이다. “공간”과 “시간”의 리듬감, 느슨함과 조임, 가벼움과 무거움, 빠르고 느림 등등의 요소를 갖는다. 다양한 서체는 개성적인 서예가의 작품을 통해 각기 다른 필선으로 표현되고 이 과정에서 리듬감의 강약도 자연스레 달라진다. 전서, 예서, 해서, 행서, 초서 중에서도 초서의 리듬감이 가장 뛰어나며 리드미컬한 움직임의 전환도 가장 빠르게 이루어진다.

---

33) 墨分五色, 묵분오색.

오늘날에는 “뉴미디어”의 대중화가 이루어지면서 “문자”의 물리적인 특징이 스크린 안으로 옮겨지고 있다. 한자 필선의 아름다움에 대한 분석을 진행하여 도출한 특징을 “키네틱 한자”에 응용하면, 우리는 추상적인 선을 조합하는 방식으로 키네틱 한자의 아름다움을 구현해낼 수 있다. 이에 따라 제5장 마지막 부분에 “선”을 주요 컨셉으로 삼아 창작한 키네틱 한자 작품을 제시하였다.

5.13 <冬節> : 동절

해당 작품은 “보이지는 않지만 볼 수 있는” 키네틱 한자 작품을 창작하고자 하는 취지에서 만들어졌다. 창작 영감은 다른 사람 손바닥에 글자를 쓰는 것에서 얻었다. 비록 문자가 남아있지 않아 어떠한 형태도 볼 수 없지만, 아래의 그림 46,47처럼 양측은 모두 동작을 통해 무슨 내용을 전달하려는지 잘 알 수가 있다.

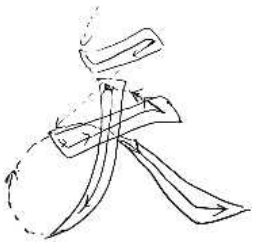


Figure 46.

Figure 47.

한자의 선은 보통 한 폭의 서화, 한 단락의 문장, 한 권의 책, 한 페이지 안에서 평면으로 존재한다. 우리가 보는 최종 형태는 움직이지 않고 정지되어 있는 것이다. 사실상 이러한 2차원 세계의 선은 3차원 공간에서 손으로 그려낸 궤도(線路, 선이 지나간 길)로 이해할 수 있는데, 우리는 흔히 이러한 궤도를 간과한다. 이러한 궤도를 어떻게 움직임으로 표현하느냐, 이것이 바로 해당 작품에서 연구한 핵심 내용이다.

궤도를 표현할 때 주의해야 할 것은 스크린 위에서 움직이는 궤도가 시각적으로 머무르는 시간이 짧아 우리가 모든 궤도를 재빠르게 볼 수는 없다는 점이다. 그런데 한자는 다른 문자와 달리 글자마다 고유한 필획 순서를 갖고 있으며 중국인은 한자를 학습할 때 이러한 필획 순서로 글자를 기억하는 특징이 있다.

이해를 돕기 위해 한자의 "오얏 리(李)" 자를 실례로 들어 위의 내용을 보충해보겠다. 아래 그림 48은 프로세싱(processing)을 사용하여 "李" 자의 움직이는 궤도를 만든 것이다.

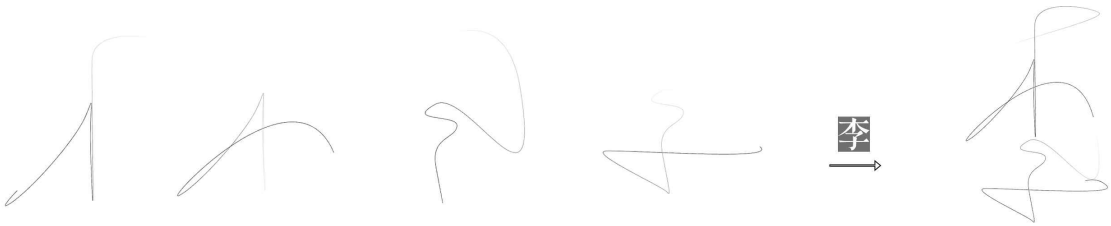


Figure 48."李" 자의 순간적인 시각적 머무름

SVG 와 CSS를 통해서 우리는 물체가 SVG 루트를 따라 움직이는 효과를 만들어 낼 수 있다. SVG 애니메이션은 일반적으로 stroke-dasharray와 stroke-dashoffset 두 가지 속성으로 구현된다. 해당 작품의 제작 원리는 self defining한 SVG 루트를 path 코드로 바꾼 것으로, SVN 화소 포인트 좌표를 얻은 다음 다시 프로세싱(processing)을 사용해 follow 루트 효과를 완성했다.

이와 동시에 작품의 정취를 더하기 위해 해당 작품의 완성 시기와 전시 시기를 모두 겨울로 선택하는 사항도 고려하였다. 이에 "동절(冬節)" 두 글자를 채택하여 재창작을 진행했다. "동절" 즉 "동지"는 24절기 중 중요한 절기이자 중국 민간에서 지내는 전통 세시풍속이기도 하다. 작품에 동지를 사용할 때 중국 각지에서 먹는 음식을 선(線) 궤도로 대체했고, 일정한 공간 안에서 시간의 흐름에 따라 무한하게 이동하는 입체적인 "동지"를 표현했다.

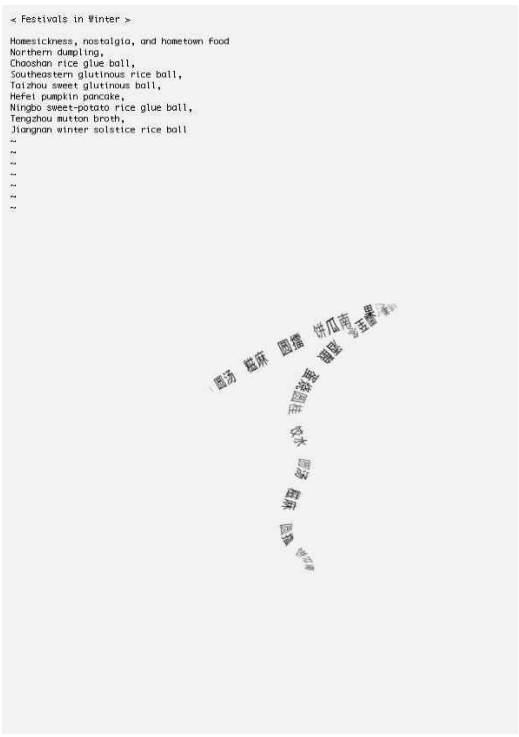
작품 감상 에세이:

《冬節》
思鄉情，家鄉飯
北方水餃 ，潮汕湯圓
東南麻餈，台州擂圓
合肥南瓜餅，宁波番薯湯果
滕州羊肉湯 ，江南冬至圓
蘇州釀酒 ，嘉興桂圓燒蛋

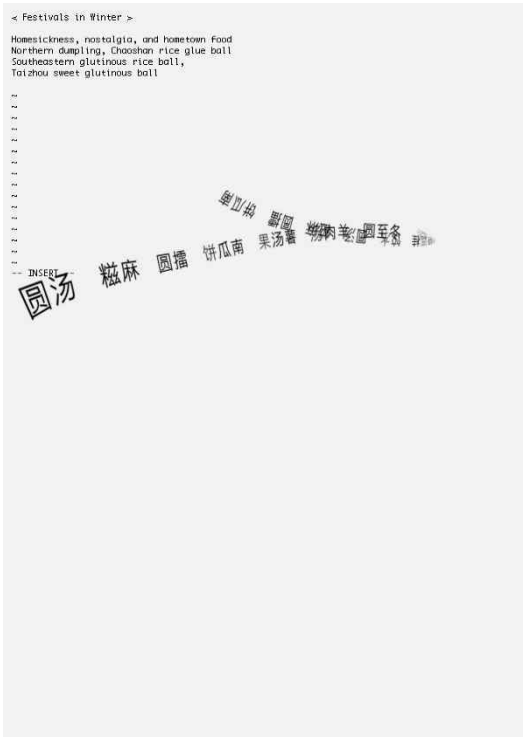
<동절>
시골의 정, 고향의 음식이 그립네
북방의 물만두, 조산의 새알 죽 탕원(湯圓)
동남의 검은깨 떡 마자, 태주의 콩가루 떡 뢰원(擂圓)
합비의 호박병, 영파의 고구마죽 번서탕과(番薯湯果)
등주의 양고기탕, 강남의 동지 떡 동지원(冬至圓)
소주의 잘 익은 술 양주(釀酒),
가흥의 대추과육 계란탕 계원소단(桂圓燒蛋)

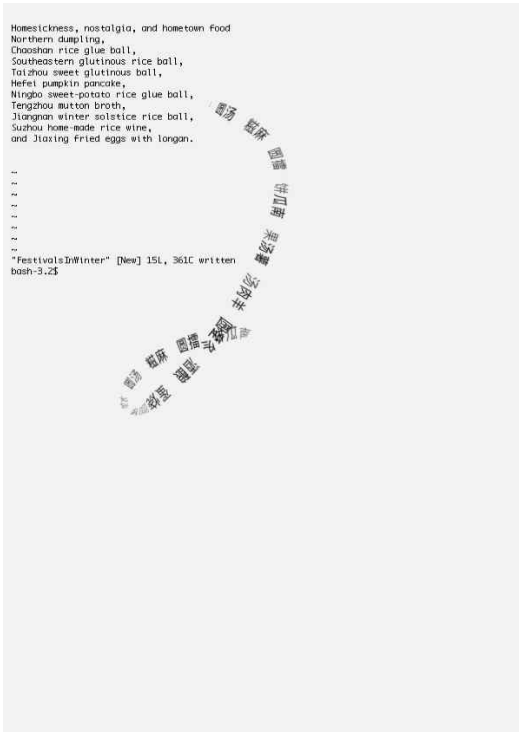
<Festivals in Winter>

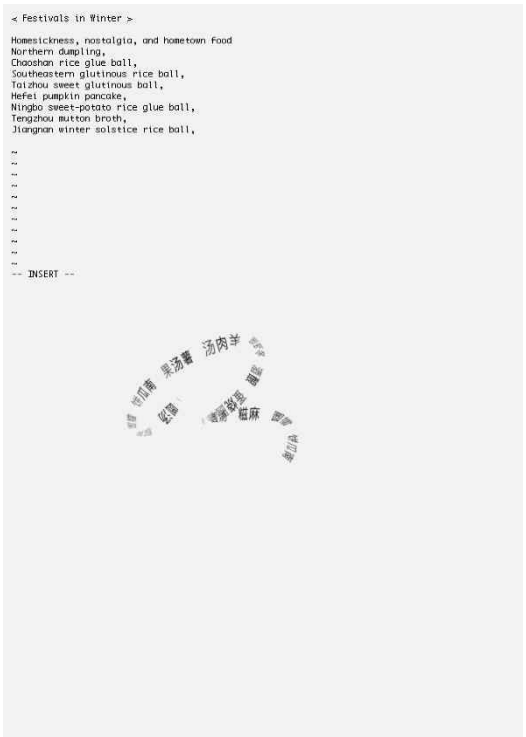
Homesickness, nostalgia, and hometown food
Northern dumpling, Chaoshan rice glue ball
Southeastern glutinous rice ball, Taizhou sweet glutinous ball
Hefei pumpkin pancake, Ningbo sweet-potato rice glue ball
Tengzhou mutton broth, Jiangnan winter solstice rice ball
Suzhou home-made rice wine, and Jiaxing fried eggs with longan

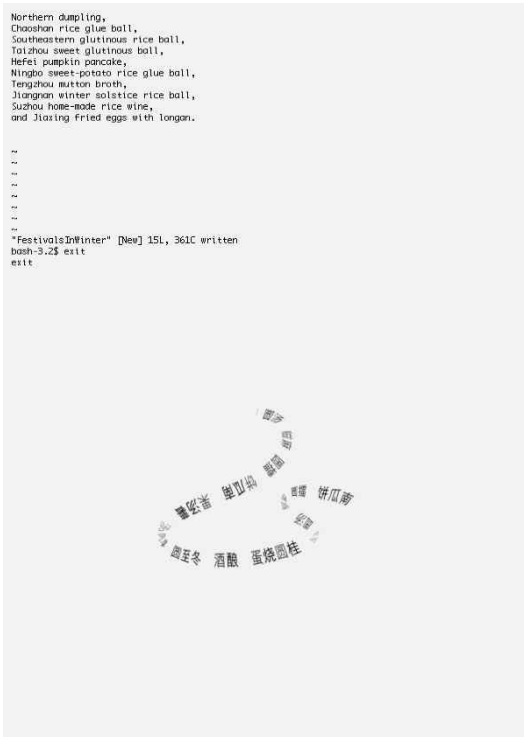


☑다차원 ☑형태의 시각적 시점 ☑기호화 ☑ 표의성 ☑은유성









☑다차원 ☑형태의 시각적 시점 ☑기호화 ☑ 표의성 ☑은유성

5.14 <人心> : 인간의 마음

선은 물체를 구성하는 시각 이미지의 기본 요소이자, 서예 예술 형식의 유일한 수단이기도 하다. 그러나 한자에서 선은 결코 기계적인 선이 아니다. 감정이 묻어나는 선, 마음을 따라 그리는 선이다. 한자는 사람의 마음과 정서 그리고 쓰는 동작 등은 서로 달라 다양한 선이 만들어진다. 똑같은 하늘 천(天)" 자라도 사람에 따라 다르게 표현된다. 아래 그림 49는 서예가가 희로애락의 감정을 몸의 움직임을 통해 가시적인 "초서"로 표현한 것이다.

Figure 49. 하늘 천자 초서

한자를 쓰는 과정에서, 어떤 선들은 어느 한 곳에서 "방향을 바꾸는" 랜덤 속성이 매우 강하다. 개성적인 필선으로 표현하려면 개별 선의 독립적인 아름다움을 따져보아야 한다. 이것은 사실 매우 모순적이다. 한편으로는 글자 형태 구조의 제약에서 벗어나야 하고, 다른 한편으로는 형

상을 그려내야 하기 때문이다. 이러한 과정은 우리에게 수학적인 “선 교차점의 해 구하기(finding a normal point)”를 연상하게 한다. 개별 선은 모두 자기만의 방향 전환의 힘(Steering Force)을 가지고 있어 독립적으로 도달하고자 하는 점을 찾을 수 있다.

이에 따라 선의 종류가 아무리 다양하더라도 최종적으로는 한자 외형의 윤곽과 필획 순서가 비슷해진다. 이러한 특징에 의거하여 <人心> 작품은 크레이그 레이놀즈(Craig Reynolds)의 Path Following 알고리즘을 사용하여 먼저 대략적으로 나아갈 방향을 정하고 그다음에 여러 갈래의 패스[34]를 진행했다.

“초서” 체는 한자를 단순한 쓰기 방식에서 예술적인 표현이 가능한 형태로 발전시켰다. 이에 <人心> 작품에 초서 서체를 사용하여 <人心> 두 글자를 표현하였다. 한정적인 범위 안에서 밀집한 형태의 가는 선을 생성하여 붓의 터치를 만들어내고, 선이 나아가는 방향을 미리 계산하여 서체 순서를 그대로 담아냈다. “다양한 인간의 마음”과 “마음의 복잡한 뒤엉킴”을 표현하기 위해 두 개의 글자를 교차하여 중첩하였다.

34) “Path Following with Multiple Segments”. Daniel Shiffman - <THE NATURE OF CODE>. p297.

작품 감상 에세이:

《人心》

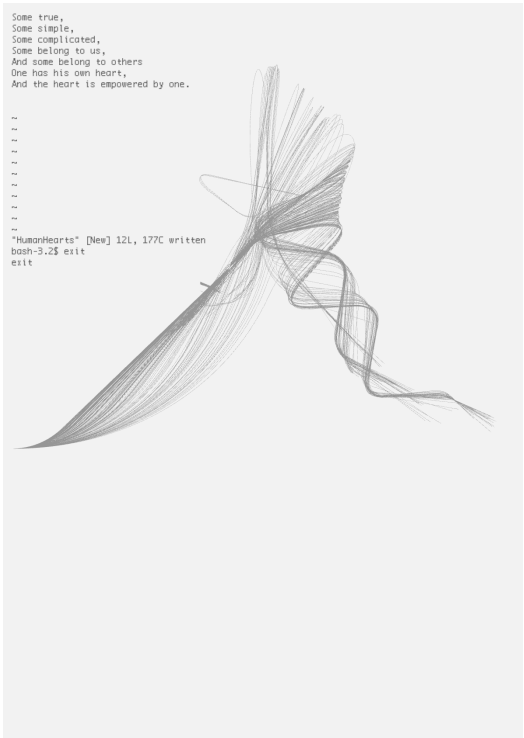
人心有“虛”“實”之分
人心有“簡”“繁”之分
人心有“你”“我”之分
人人有心，心發自人

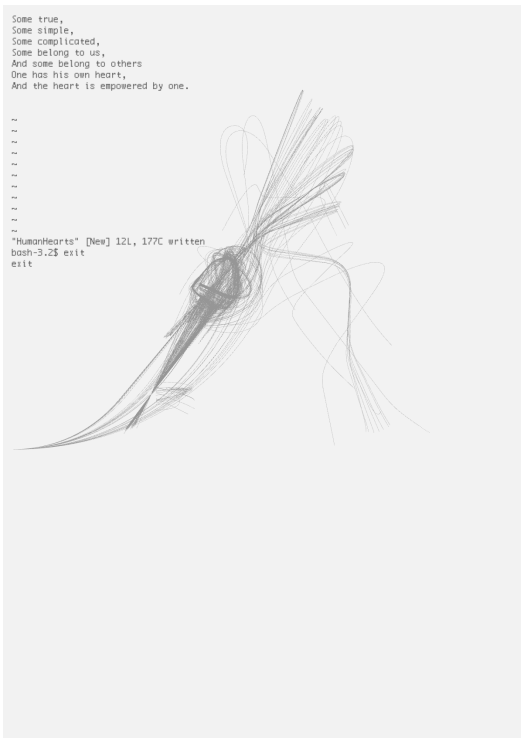
<인간의 마음>

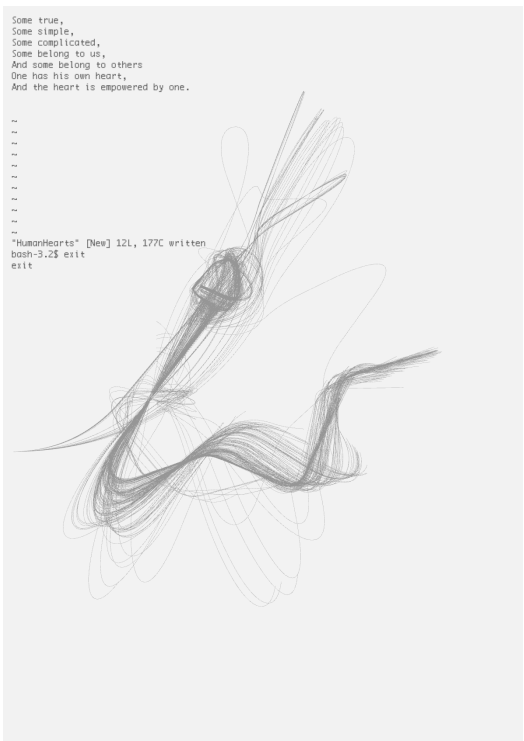
인간의 마음은 “허상”과 “실체”로 구별되고
인간의 마음은 “단순함”과 “복잡함”으로 구별되며
인간의 마음은 “너”와 “나”로 구별된다
누구나 이러한 마음을 가지고 있으며,
이 마음은 인간으로부터 시작되었다.

<Human Hearts>

Some false, some true
Some simple, some complicated
Some belong to us, and some belong to others
One has his own heart, and the heart is empowered by one







☑다차원 ☑복합성 ☑형태의시각적 시점 ☑기호화 ☑ 표의성 ☑상형성 ☑은유성

5.15 <看到風> : **바람을 바라보다**

한자는 상형문자를 토대로 발전하여 이루어진 문자로 상형문자는 원시 그림문자가 발전하여 완성되었다. 그런데 그림으로 그려 사물과 비슷하게 표현하려면 필획은 자연히 많아지고 쓰는 데에도 시간과 정성이 많이 들어가게 마련이다. 이런 이유로 한자는 단계별로 간략화 과정을 거치게 된다. 대전(大篆)에서 소전(小篆)으로 간략화되고, 소전(小篆)은 다시 예서(隸書)로 간략화되었으며, 해서(楷書)에서 다시 근대 한자(近代 漢字)로 간략하게 바뀌었다.

한자의 간략화로 한자는 사용 빈도가 증가되었으며, 쓰임이 광범위해지고 한층 더 과학적이고 편리해졌다. 간략화 방법에는 여러 가지가 있는데 그중에 “소리 기호”나 “형태 기호”를 생략하는 방법이 포함된다. 점점자 “點”는 “点”로 줄어들면서 그 과정에서 형태 기호 “黑”자가 생략되었고, 아버지 야 “爺”자는 "爷"로 줄어드는 과정에서 소리 기호 “耶”자가 생략되었다. 한자의 편방부수 역시 간략화되었는데 물 수 “水”자는 “氵”으로 간략화되었고, 말씀 언 “言”자는 "讠"으로 간략화되었다.

한편 간략화되는 과정에서 여러 문제도 발생했다. 예를 들면 “已”, “已”, “己”그리고 “戊”, “戍”, “戎” 등 문자의 경우 식별하는 정도가 낮아지면서 구분하기 어려워졌다. 또한 일부 간략화된 글자의 경우에는 수정을 거친 후에 글자를 만든 본래 의도가 퇴색되기도 하였다. 예를 들어 때 시“時”자는 좌변에는 형태 기호 “日”자가, 우변에는 소리 기호 “寺(si)”로 조합되는데, 간략화된 이후에 “時”자로 줄면서 직접적으로 소리를 나타내는 기능을 상실하게 된다. 위의 내용을 통해 간략화 과정이 한자를 “재구성”하는 과정이자 문자의 “아름다움”과 “실용”의 균형을 찾는 것임을 알 수 있다. 여기에서 말하는 “실용”은 의미 전달의 정확성, 형태의 간결함, 기억의 편리함을 지칭한다.

최근에는 점점 더 많은 수의 아시아 디자이너들이 “한자의 간략화”가 한자가 변화 발전하는 전체적인 추세라는 점을 인식하고 속속 한자의 구조를 생략하는 실험을 진행하여 한자의 정수를 보여주고 있다. 2017년에

는 한중일 3국이 공동으로 주최한 “100개의 바람, 100인의 바램” 아트 포스터전에서 많은 수의 대표 작품들이 나왔다. 아래에 제시된 작품들의 공통적인 특징은 모두 한자의 바람 풍“風”자를 사용하여 진행한 디자인이라는 점이다. 아티스트들은 “几”자 안에 들어가는 내용은 다른 것으로 대체하는 방식으로 작품의 의미를 전달하였다.

<看到風> 작품의 창작 영감도 여기에서 얻었다. 다만 최소한의 요소만을 사용하여 실시간으로 변하는 자연현상 “風”을 표현하는 점이 다른 작품과 구별된다.

Figure 50. 홍보 이벤트 포스터

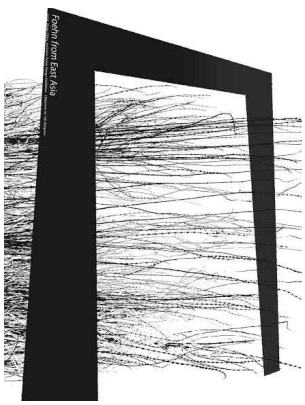


Figure 51. 부분 전시 작품
(다카하시 요시마루 (日) Title_門／문)

한자를 간략화하는 방법 35)은 일부로 전체를 표현하는 “以偏概全”이라 부르는데, 원 글자의 일부를 취하거나 혹은 글자 형태의 윤곽만 남기고 복잡한 중간 필획은 삭제하는 것을 말한다. 본인은 해당 방법을 사용하고 한층 더 “미니멀(minimal)”한 방식으로 “几”자를 추상화하였다.

그런데 “几”자로 구성되는 한자에는 “風”자 외에도 “鳳”, “凰”, “凡” 등 자도 있기 때문에 부연 설명을 추가하지 않으면 혼동하기가 매우 쉽다. 이에 해당 작품은 키네틱 한자의 디자인 기법을 적용하였고, “매우 간결(minimal)”하면서도 “구별하기 쉬운” 두 가지 조건하에서 최적의 조형 방법을 찾아냈다. “風”자의 특징을 표현하기 위해 작품에 perlin noise를 사용하여 중심에 더 위치하는 새로운 이동량을 만들고, 실선이 바람 사이로 흩어지는 느낌을 주었다.

작품 감상 에세이:

《看到風》

淨心看微風
看微風過處淡淡清香
看微風拂過絲絲溫柔
看微風吹起青春回憶
看微風穿越逝去時光
看昨日的風吹亂我心

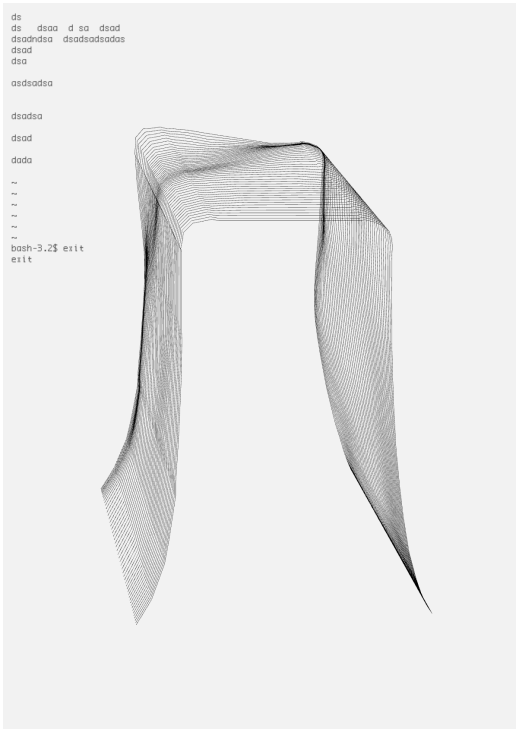
35) 1956년 중국 한자의 간략화는 아래의 몇 가지를 원칙을 따른다. 1. 원래 글자와 형태가 유사한 것으로 글자 형태의 윤곽을 남겨두거나, 원래 글자의 일부만 추출하는 법 2. “육서(六書)”의 원리에 따라 글자 형태를 개조하거나 새로운 글자 형태를 만드는 법 3. 순수 기호의 한자를 만드는 법 4. 초서와 해서, 옛 것을 발전 계승하여 현대에 맞게 적용함

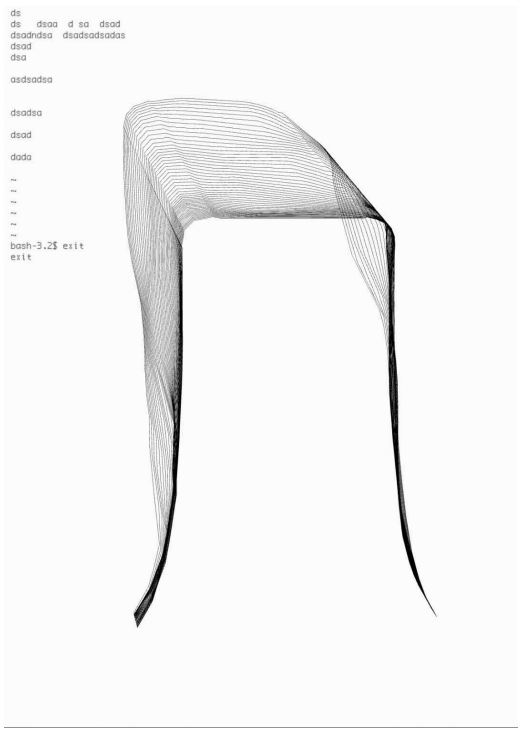
<바람을 바라보다>

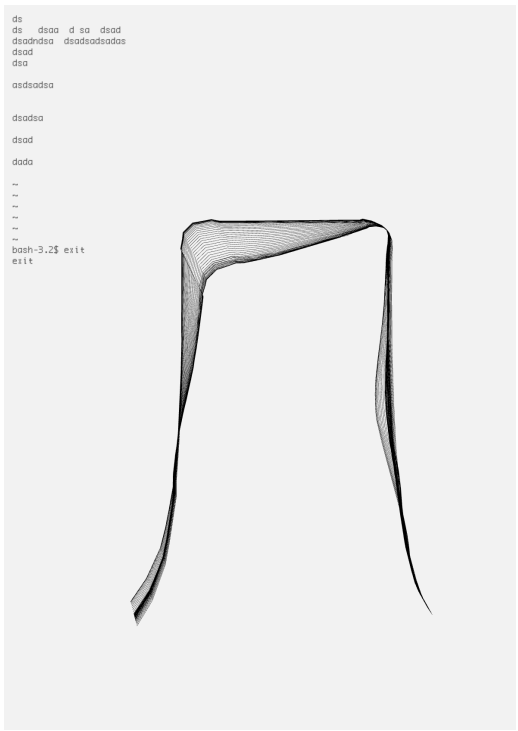
고요히 산들바람을 바라본다
산들바람이 지나가는 자리마다 맑은 향기가 일고
산들바람이 부니 은은한 부드러움이 느껴지며
산들바람 사이로 청춘의 추억이 떠오른다
산들바람으로 지난 시간을 거슬러간다
어제 불었던 바람이 내 마음을 흔든다

<Seeing the Wind>

To see the wind with a clear mind
To smell the fragrance where the wind blows
To feel the gentleness where the wind blows
To recollect the old-day memories when the wind blows
To take a journey to the lost time with the wind blowing
To see the wind yesterday disturbing my clear mind







☑다차원 ☑복합성 ☑모듈화 ☑형태의시각적 시점 ☑기호화 ☑ 표의성 ☑상형성 ☑은유성

5.16 < 無 > : 무

복잡한 형태로 이루어져 배우고 쓰기가 쉽지 않은 한자를 간략하게 만들고자 하는 노력은 송나라 때부터 시작되었다. 본격적으로 한자를 간소화하여 쓴 것은 1950대 이후의 일이다. 1951년 중국 정부는 전통적으로 써오던 한자인 “번체자(繁體字)”의 형태를 간략화하는 사업을 진행하였다.[36] 간체자(簡體字)는 예전부터 내려오던 복잡한 형체의 한자를 간략한 형태로 새로 만든 글자이다. 매우 재미있는 현상은 간체자(簡體字)만 배운 중국 젊은이들의 대부분이 번체자(繁體字)도 읽을 수 있다는 점이다. 이는 간체자가 번체자의 틀 위에서 만들어진 까닭이다. 본인은 이러한 현상이 발생하게 된 요소를 다음과 같이 분석하여 정리해보았다.

1. 한자의 조자(造字) 방법과 상호 연관
“육서” 에는 한자의 구성원리를 상세하게 다룬 내용이 담겨있는데 그 중 “형성자(形聲字)”의 경우 우리는 독음을 나타내는 부속요소로 글자를 식별한다. 이를테면 번체자 양식 양(糧)자는 간체자 “粮”으로 사용한다. 또는 “상형(象形)의 특징”을 이용하기도 하는데, 표의(表意)적 부분을 간략하게 사용해 복잡한 필획을 대체하기도 한다. 이에 해당하는 예로는 눈물 루(淚) 자를 간소화 한 “泪”자, 티끌 진(塵) 자를 간소화 한 “尘”자 등이 있다.

2. 특징적 부분으로 전체를 구성하는 간체자. 본래 글자가 여러 부분으로 구성되어 있으나 현재에는 가장 특징적인 부분만 남겨 사용한다. 번체자 의원 의(醫) 자는 간체자 “医”으로 쓰인다.

3. 윤곽으로 글자를 구성하는 간체자. 본래 글자의 윤곽을 취해 전체를 표현하는데 그 예로 새 조(鳥) 자를 들 수 있다. 간체자는 “鳥”로 쓰인

36) 1951년 중국 정부는 전통적인 한자인 번체자를 간략하게 줄이는 연구를 본격적으로 시작하여 1956년에 정식으로 간체자를 공표한다.

다.

4. 간체자는 번체자의 필획이 생략되어 만들어졌다. 본래 글자가 획이 많거나 복잡한 경우 간략하게 기호화 한다. 대쪽 간, 간략할 간(簡)자는 간체자 “簡”으로, 지나갈 과(過)자는 간체자 “過”으로 쓰인다.

5. 간체자의 편방(偏旁)으로 번체자를 추측할 수 있다. 간략화 한 편방 부수로 합성된 글자를 추측할 수 있다. 그 예는 다음과 같다. 편방자 금(釒)을 간체자 “钅”으로, 물고기 어(魚)자를 간체자 “魚”으로 줄여 사용한다.

6. 번체자를 사용하는 지역과의 빈번한 교류가 있었기 때문이다. 장기간 홍콩, 마카오, 대만의 영향을 받은 영화나 TV 프로그램의 자막과 문학 작품을 그 예로 들 수 있다.

7. 필기체의 영향을 받은 요소도 이에 포함된다. 초서(草書)체 중에는 극도로 간소화 되어 쓰는 방법이 있는데 이것이 통용된 경우를 그 예로 들 수 있다.

간체자는 중국인의 일상적인 교류와 학습에 편리함을 가져다주지만 브랜드, 광고, 잡지 등의 평면 디자인 중에는 번체자를 사용하여 창작한 경우가 더 많다. 번체자 안에는 상당한 양의 원시정보가 담겨 있으며 글자 하나하나가 마치 그 내용을 함축하고 있는 이미지와도 같고 디자인 할 수 있는 영역이 더 많기 때문이다. 이 같은 이유로 디자인 분야에서 특히나 번체자가 디자이너의 사랑을 더 받는 것이다.

“무(無)” 자를 주제로 삼은 두 가지 이유:

첫째, 한자를 기술과 결합할 때 흥미로운 공감대가 형성된다. 프로세싱

을 배운 사람이라면 입자(Particle)로 물리 현상을 시뮬레이션 할 때 오픈소스(open-source)가 많이 있다는 걸 알고 있을 것이다. 우리는 기존의 물리함수 피직스 라이브러리(Physics Libraries)를 사용해서 원하는 대로 가상 물리 세계를 구축할 수 있다. 본인은 Box2D와 toxiclibs (VerletPhysics)을 사용해 유연성을 가진 물체를 시뮬레이션 해보았다. 그림 52 에 있는 장력을 가진 격자가 바로 그 예이다. 탄성을 가지는 격자를 통해 보는 이에게 사각형과 관련된 사물을 연상시킬 수 있는데 본인은 이를 한자의 “무(無)”자와 함께 연결 지어보았다.

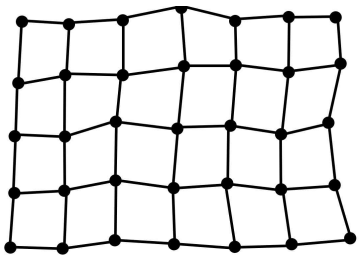


Figure 52. 유연한 사각형

둘째, 본 연구자의 문화적 배경과 관련된 이유에서다. 누군가는 분명 왜 주제를 “그물 망(网)”이나 “밭 전(田) “자를 택하지 않고 “없을 무(無)”자를 선택했냐고 물을 것이다. 이는 중국의 전통문화인 “도교”, “불교”, “유교”의 3대 종교문화에서 비롯된 연유다. 3대 종교 안에는 모두 “무(無)”와 관련한 사상과 연계되어 있다. “무위(無爲)”, “무극(無極)”, “무상(無想)”, “무아(無我)” 등을 그 예로 들 수 있겠다. 또한 다수의 아시아 예술 작품들은 “무(無)”로 예술의 경지를 표현하기도 한다. 변화무쌍한 묘경을 나타내는 공영(空靈)이나 아무 것도 없이 텅 비어있는 허무(虛無) 등의 개념을 무(無)로 표현한다. “무(無)”자는 아시아에서 없어서는 안 될 중요한 사상적 “기호”이다.

셋째, 고대 한자에서 “없을 무(無)”자와 “춤출 무(舞)”자는 하나의 글자로 쓰였다. 고대 원시인들의 춤은 단순히 즐기는 것에만 국한되지 않는다. 춤 안에는 신으로 받들어지는 영혼 또는 자연물을 섬기는 동작의

의미도 담겨있다. 신은 보이지 않고 손으로도 만질 수 없으며 이야기를 할 수도 그 형상을 알 수도 없다. 인간은 춤을 출 때 하늘의 존재를 상상하였다. 이런 이유로 “무(無)” 자는 두 가지 의미를 갖게 된다. 무(無)” 자는 춤으로 표현하는 가시적인 동작 그 자체와 보이지 않는 대상을 표현하는 동작으로 사용되었다. 이러한 점에 착안하여 작품의 제목을 <무(無)>로 선택하였다.

작품 감상 에세이:

<無>

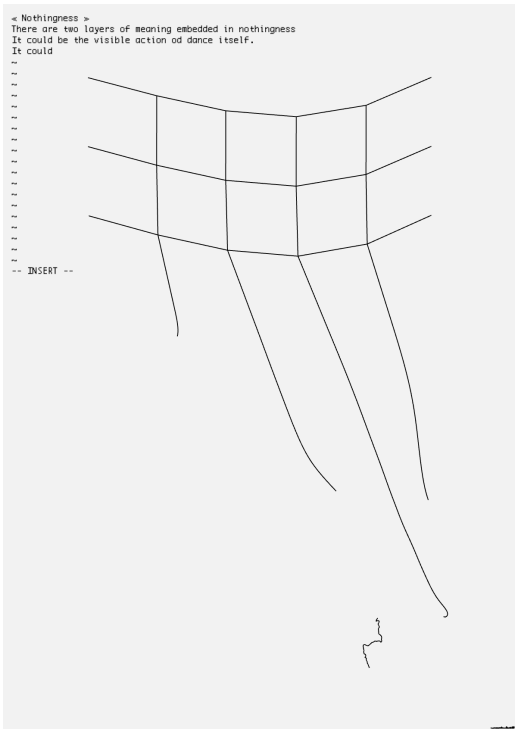
“無”有双重含義
卽可以表示舞蹈的可見動作本身
也可以表示舞者心中那个看不見的對象

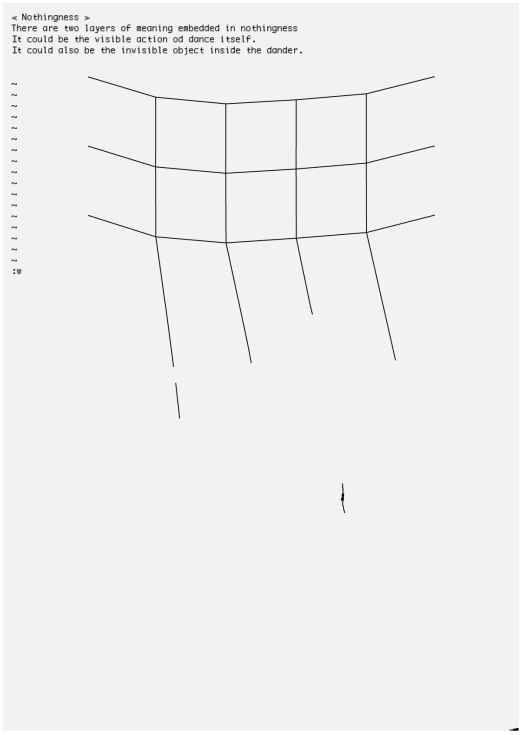
<무>

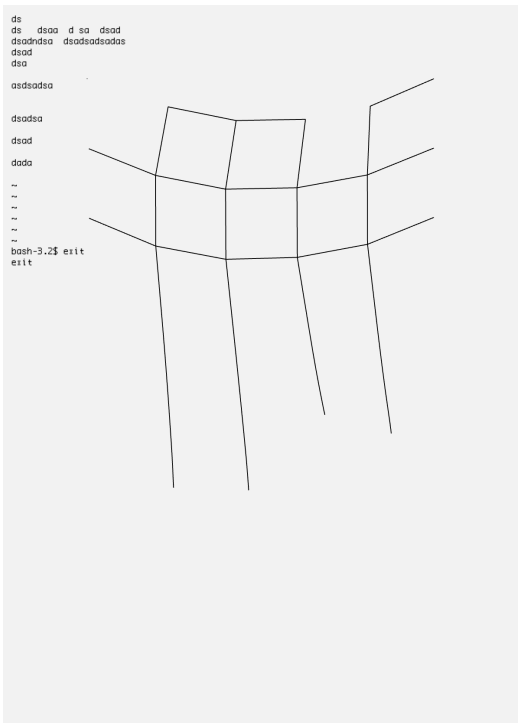
“무(無)”는 이중적 의미를 갖는다.
춤으로 볼 수 있는 몸짓 그 자체를 표현할 수 있고
춤추는 이의 마음속에 있는 보이지 않는 대상을 표현할 수도 있다

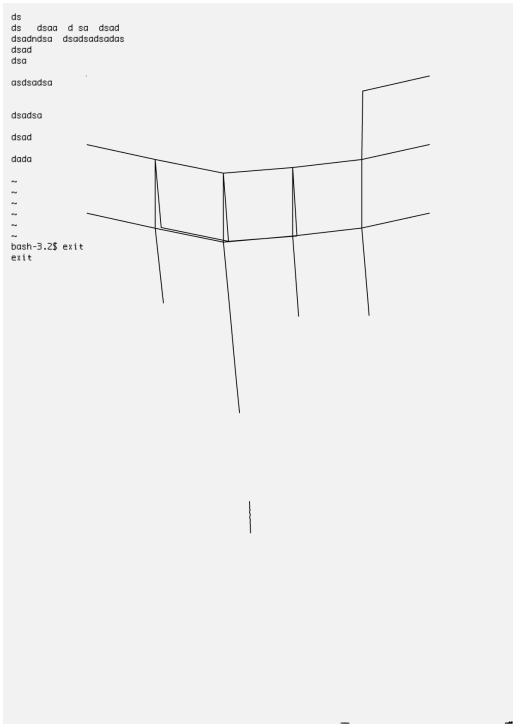
<Nothingness>

There are two layers of meaning embedded in nothingness
It could be the visible action of dance itself.
It could also be the invisible object inside the dander.









☑다차원 ☑형태의시각적 시점 ☑기호화 ☑ 표의성 ☑상형성 ☑은유성

제6장 결론

6.1 연구의 결과 및 결론

우리는 매일 다양한 정보에 둘러싸이고 각종 문자 정보는 각기 다른 방식으로 우리의 시야에 들어온다. 고전적인 읽기 방식은 “직선식 읽기”라고 말할 수 있는데 현대에 들어서 정보량이 폭발적으로 증가하면서 정보 읽기는 단순히 읽는(reader) 것에서 열람하는 서핑(browsing) 방식으로 바뀌었다. 이처럼 사물을 바라보는 시각(視覺) 문화의 구조적인 변천이 이루어지며 문자는 점차 “유동적이고 변화하는” 방식으로 우리의 시선을 사로잡고 있다.

타이포그래피 역사를 살펴보며 우리는 전체적인 디자인의 트렌드가 새롭게 바뀌고 있음을 알 수 있다. 문자 디자인에 시공간이 결합하면서 문자는 점점 더 복잡하고 정교하게 활용되고 있다. 디지털 시대에 들어서면서 텍스트(text)의 공간(space)은 한층 더 유동적이고 비고정적으로 바뀌었으며 타이포그래피 또한 고정적인 실체(實體)에서 다양하게 변화하는 형태(形態)로 진화하였다. 이와 같은 움직임을 만들어가는 과정에서 타입(type)은 “무빙(moving)”, “모션(motion)”, “다이내믹(dynamic)” 등 용어와 쉽게 혼용되어 사용되는데 본 논문에서 지칭하는 키네틱 타이포그래피 용어는 이러한 어휘들을 모두 포함하여 통칭하고 있음을 밝힌다. 근래에 창작된 적지 않은 수의 키네틱 타이포그래피 우수작은 “시간과 움직임”의 특질을 가지고 있으며 동시에 컴퓨터 알고리즘과 IC 칩 등 신기술을 작품에 접목하는 양상을 보이고 있다. 이에 본 논문에서 4단계로 나누어 한자 키네틱 타이포그래피의 발전 가능성을 구체적으로 살펴보았다. 그 내용은 다음과 같다.

첫 번째 단계에서는 가장 먼저 키네틱, 키네틱 아트, 키네틱 타이포그래피에 대한 개념을 정의하고 역사적인 맥락을 정리해보았다. 주지하는 바와 같이 “키네틱”은 서양에서 기원하였다. 초기에는 키네틱 아트와 가

장 직접적으로 관련되었는데 20세기 초의 회화, 조각, 설치 미술에서 키네틱과 관련된 작품을 찾아볼 수 있다. 키네틱 아트는 다음과 같은 특징을 가지고 있다. a 역동성으로 선과 미묘한 형태로 조합되어 움직임 자체를 표현 양식으로 삼아 움직임의 조형미를 추구하고, 일정한 시간의 개념이 더해지면서 움직이는 힘은 작품의 중요 요소로 작용된다. b 시지각성(視知覺性)으로 오브제의 시지각적인 움직임 또는 실제적인 움직임, 혹은 관찰자의 움직임에 의해 긴장감이나 특수 효과를 얻을 수 있는 특성이다. 오브제의 움직임에 의한 커다란 시각적 효과와 더불어 착시적인 효과, 관찰자가 보는 위치에 따른 형태의 변화 등 다양한 효과를 생성하는 시지각성은 작품의 의의를 높이며 또한 시간에 따라 형태가 변형되면서 관찰자의 시지각을 자극하고 주의를 집중시키는 특징을 보이기도 한다. c 변형성으로 오브제의 빠른 움직임에 의해 형태가 변형되어 보이거나 비물질화되는 속성을 일컫는데 대상이나 관람자의 움직임에 따라 작품 자체에 다양한 변화가 생기면서 미적인 요소를 갖는 것을 말한다.

키네틱 타이포그래피는 키네틱 아트의 여러 조형적인 특징을 계승하였으며 “문자”를 주요 창작 요소로 삼는다. 3장에서는 키네틱 타이포그래피의 디자인적 특징을 다차원적 특징, 운동적인 특징, 기술의 다원화로 나누어 살펴보았다. 여기에서 말하는 “다차원”이란 주로 3차원과 4차원적인 공간을 대상으로 디자인한 것으로 시간과 공간에 대한 탐색, 소리, 촉감 등이 그 예이다. “운동적 특징”은 자연현상, 물리적 상태, 행위, 감정, 리듬 등을 시뮬레이션 하는 것을 포함한다. “기술의 다원화”는 하드웨어의 확장, 양방향성(interaction) 등을 키네틱 타이포그래피의 발전 트렌드로 삼았다.

두 번째 단계에서는 “한자”에 대한 연구를 진행하였다. 키네틱 타이포그래피는 서양에서 지속적인 발전을 거듭하였는데, 토양이 다른 아시아의 생태적 환경에서 새롭게 발전시키기 위해서는 먼저 한자에 대한 심도 있는 연구가 이루어져야 하고 이를 토대로 “움직이는 한자”의 디자인 기법을 도출하는 과정이 필요하다고 생각했다. 최근 몇 년 사이에 “형태(形)”와 “의미(意)”가 한자 디자인의 새로운 소재로 활용되고 있다. 몇몇

2차원 디자인 작품은 한자의 “형태”와 “의미”를 시각적인 특징으로 활용하여 매우 독창적인 시각 표현방식으로 한자 디자인을 표현하고 있다. 다수의 디자인이 증명하듯 한자의 특징을 더 많이 발굴해야만 한자에 담긴 정보를 보다 더 정확하게 응용할 수 있을 것이다. 한자를 식별하는 과정은 이미지를 통해 완성되는 시각적인 사유 과정이자, “구체적인 한자 모양”과 “추상적인 한자 의미”를 고도로 하나로 통합하는 과정이기도 하다. 이러한 특징들은 키네틱 한자 디자인 과정에서도 반드시 구현되어야만 한다. 키네틱과 관련된 모든 문자는 운율성, 연속성, 시간성, 역동성 등의 공통적인 속성을 갖고 있다. 이에 본고는 키네틱 한자의 디자인적 특징을 구조적 특징, 기호와 상형의 특징, 표의와 은유성의 특징으로 구체적으로 나누어 살펴보았다.

세 번째 단계에서는 작품 케이스에 대한 분석을 진행하였다. 유사한 사례와 비교 분석하는 작업은 노하우를 축적하는 데에 도움이 되어 향후 작품 창작의 기반이 될 수 있다. 한자와 관련된 키네틱 작품을 창작할 때 현대적인 요소를 가미하면 대중의 심미적인 취향을 만족시킬 수 있으므로 디자이너는 변하지 않는 전통적인 스타일만 계속 고수할 필요가 없다. 키네틱 한자 디자인의 발전을 위해 해결해야 할 과제도 있다. 현재 키네틱 한자 디자인은 두 가지 난제를 가지고 있다. 하나는 한자의 디자인적 특징을 추출하는 것이고 다른 하나는 시각적인 효과를 표현하는 적합한 기술 수단을 모색하는 것이다. 빼어난 키네틱 한자 작품을 창작하는 것은 결코 쉬운 일이 아니다. 오랜 기간에 걸친 디자이너의 훈련과 학습이 뒷받침되어야 한다.

마지막 단계에서는 이러한 연구 내용들을 토대로 작품을 창작해보았다. 작품은 모두 코드, 디지털 모형, 알고리즘 등을 기반으로 창작한 것들이다. 인류의 순수 이성으로 창작한 예술은 컴퓨터의 시각화 과정을 통해 우리에게 현대 과학기술이 선사하는 무한한 매력을 보여주고 있다. 일론 머스크(Elon Musk)가 말한 바대로 “테크놀로지가 가장 마법에 가깝다”.

“논리”에 “감성”의 힘을 불어넣어 창작을 진행하는 일은 마치 물리적

인 경계에 무한한 생명의 힘을 부여하는 것과 같다. 이에 본인은 동양인의 감성과 한자의 오묘한 매력을 바탕으로 컴퓨터와 수학적인 공식을 캔버스 도구로 삼아 작품을 창작했다. 그리고 5장에는 개별 작품마다 감상 에세이를 덧붙였다. 이는 서양의 창작자가 종종 작품 감상 에세이를 통해 관객과 상호 교류하는 데에서 연유한 것이다. 그 밖에도 아시아의 전통 예술에는 “시화동원(詩畵同源)”이라는 개념이 있는데 시(詩)는 화(畵)의 기초이자 필요조건이고, 화(畵)는 시의 의미가 그림의 모양이나 형태로 전환된 예술이라는 뜻이다. 이미지를 표현하지 않고 언어나 문자로만 시(詩)의 의미를 전달하는 이는 불완전한 시인이라 할 수 있고, 그림에 시(詩)적 의미를 표현해내지 못하는 이는 고작 기술에만 능한 ‘화공畵工’에 그친다고 말할 수 있다. 작품이 시적인 정취와 그림 같은 아름다움을 모두 갖추고 있을 때 대중의 인정을 받게 마련이다. 이런 까닭으로 부족하게나마 디자인 작품에 감상 에세이를 덧붙여보았다.

단계별로 살펴본 이상의 내용이 본 논문의 결론이다. 본 연구자의 키네틱 한자에 대한 연구는 이제 막 시작되었다. 아시아 문화에 뿌리를 둔 동양인의 미학적인 관념으로 한자 디자인을 바라보니 그 안에 앞으로 개발할 “디자인 창의(creative)성”이 무수히 많이 존재하고 있음을 발견할 수 있었다. 먼지가 켜켜이 쌓여 오랜 시간 창고에 보관되어 있던 문화적 자원은 오늘날 다시 커다란 보물창고로 거듭나고 있다. 이에 본 연구자는 동양 문화의 정수인 “한자”를 작품 창작의 돌파구로 삼아 멀티미디어 시대의 과학기술과 융합하여 디자인의 새로운 가능성을 열어나가겠다.

6.2 연구의 한계점 및 향후 연구 과제

이미 많은 수의 사람들이 동양의 문화적 자원을 발굴하고 활용하고 있지만 전통문화를 그대로 복제하는 행위는 우리 사회에서 환영받지 못하고 점차 도태되고 있다. 앞으로는 시대와 함께 발맞추어 나아가는 창작이 필요하다. 전통과 현대를 완벽하게 결합하는 작업이 결코 쉬운 일이 아니다.

논문에서 언급하는 프로그램은 이를 돌파하기 위한 하나의 창작 수단이다. 여타의 인터페이스의 컴퓨터 소프트웨어와 다르게 사용 시 소프트웨어의 자체적인 제약을 받지 않는다. 아이디어만 있으면 맞춤형 알고리즘을 구성하여 자신이 원하는 바를 표현할 수 있다. 이는 예술 창작에 새로운 요소를 더하는 것이자 또한 창작자의 시야를 한층 더 확장시켜주는 것이기도 하다. 다만 아쉽게도 실제 작업 과정이 매우 복잡해 해당 분야의 전문적인 지식을 필요로 한다는 점이다. 새로운 알고리즘을 짜거나 프로그램 언어를 고도로 능숙하게 활용하는 작업은 예술 분야에 종사하는 아티스트에게 있어 매우 어려운 일이 아닐 수 없다.

논문의 최종 작품에서 활용한 소프트웨어와 관련된 알고리즘과 기술 등은 이미 수년 전부터 광범위하게 사용되는 것들이다. 알고리즘을 사용해 구축한 운동 경로와 화려한 효과들도 이제는 모두 빅데이터, 머신러닝, AI를 활용하여 구축할 수 있다. 다만 예술 분야에 종사하는 이들에게 어떻게 가르칠 것인지, 학습 비용을 어떻게 낮출 것인지, 기술적 제약을 어떻게 극복할 것인지 등의 내용은 새로운 예술 창작을 위해 앞으로 계속 연구해야 할 과제들이다. 연구가 제약적이고 부족한 점이 많지만 적게나마 본 논문이 향후 키네틱 한자 연구에 도움이 되길 바란다. 앞으로도 본인은 한자에 대한 연구를 지속하여 새로운 키네틱 한자 작품을 창작하며 관련 분야의 발전을 뒷받침해 나가겠다.

제작환경

컴퓨터

Retina, 15-inch, Mid 2015

2.2 GHz intel Corei7

16 GB 1600 MHz DDR3

Intel Iris Pro 1536MB

소프트웨어

Processing

PCtolCD2002

Sublime

Photoshop

Illustrator

한글

웹호스팅 환경

리눅스 웹 호스팅

Linux

Apache

mysql5

PHP

참 고 문 헌

단행본

엘런 럽튼, 김주성, 스크린 타이포그래피, 비즈앤비즈, 2015

신청우, 디지털타이포그래피, 임프레스, 2003

원유홍, 서승연, 타이포그래피 천일야화, 안그라픽스, 2004

Rickey George, 윤난지. 키네틱 아트 Kinetic art, 1988

이현영, 웹&모션 타이포그래피의 정수, 정보문화사, 2011

김동빈, 타이포그래피 미학, 커뮤니 케이션북스, 2015

Daniel Shiffman, The nature of code, 2012

Barbara Brownie, Transforming Type (New Directions in Kinetic Typography). Bloomsbury. 2015.

Daniel Shiffman, Learning Processing - A beginners Guide to Programming Images,Animation, and Interaction Second Edition, China Machine Press, 2017

Laszlo Moholy Nagy, Vision in Motion, CHINACITICPRESS, 2016

Jon Krasner, Motion Graphic Design, applied history and aesthetics, Focal Press, 2016.

Lothar Ledderose, Ten Thousand Things-Module and Mass Production in Chinese Art, Princeton University Press, 2011

劉又辛, 方有國, An Outline of the History of the Development of the Chinese Characters, 中國大百科全書出版社 2000

柯志杰, 蘇煒翔, Type face 산책-字型散步, 臉譜, 2014

傅永和 , 漢字七題, 河南教育出版社, 1993

Mark Sanders, Sandra Maxa, Ben Day,Philip B.Meggs, Rob Carter, Typographic Design: Form and Communication, 6th Edition. 2014,

논문

이지은, 노사카 마사시, 키네틱 타이포그라피와 인쇄매체 타이포그라피의 감성 구조의 차이, 한국타이포그라피학회,993~1176

노사카 마사시, 문자에서의 미적 가치 판단 기준-시각시와 서예의 비교를 중심으로, 한국타이포그라피학회,993~1176

오재희, 김지현, 오브제티미지 실험 연구 - 영상과 오브제를 활용한 키네틱 타이포그래피, 한국타이포그라피학회, 737~992

윤난지, 움직이는미술에 관한 연구- 확장된 작품개념을 중심으로, 이화여자대학교 대학원, 1991

김병조, 타이포그래피 공간의 구조, 홍익대학교 대학원, 2012

박한수, 타이포그라피 공간감을 통한 텍스트의 구체화 연구, 홍익대학교 대학원, 2009

이준환, 김동환, 위지은, 장수연, 하세용, 전수진, 키네틱 타이포그래피를 통한 텍스트 기반 커뮤니케이션에서의 감정 전달 연구,멀티미디어학회 논문지 제17권 , 2014

Frank Popper, l'art cinétique. Einaudi Turin, 2005

Sooyeon Lim, Design of Kinetic Typography Interaction based on the Structural Characteristics of Hangul, Dongyang University, 2016

Johnny C. Lee, Jodi Forlizzi, Scott E. Hudson , The Kinetic

Typography Engine: An Extensible System for Animating Expressive Text, Carnegie Mellon University, 2002

Ford, S., Forlizzi, J., and Ishizaki, S. “Kinetic Typography: Issues in time-based presentation of text. Conference Extended Abstracts,1997

Teemu Ikonen. Moving text in avant-garde poetry -Towards a poetics of textual motion, 2003,

Theo van Leeuwen, Emilia Djonov, Research paper- Notes towards a semiotics of kinetic typography, Social Semiotics, 2015.

Yin Yin Wong, Temporal Typography - Characterization of time-varying typographic forms, Carnegie Mellon University, 1995

웹 사이트

namu.wiki/w/물리학
ko.wikipedia.org/wiki/정역학
ko.wikipedia.org/wiki/운동학
en.wikipedia.org/wiki/Kinetic
ko.wikipedia.org/wiki/역학
www.britannica.com/science/kinetics
tinganho.info/Motion-Type-Project
blog.logo123.net/23810
www.youtube.com/watch?v=IyN8Ga3CdQU
typojanchi.org/2015/ko/1-doosup-kim
www.cmu.edu/cfa/design/kdg/kt/
www.mk.co.kr/news/society/view/2017/10/677234/

부록

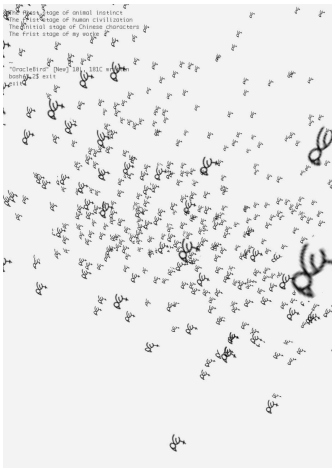
(5장에서 분석한 창작물과 전시 현장사진은 www.xinlin.art를 방문 하거나 lxl.snu.ac.kr 에서 확인할수 있다)

작품의 기술적인 부분 설명

개인 작품 왼편에는 모두 작품에 해당하는 코드/감상 에세이를 달았다. 감상 에세이의 움직이는 효과는 ttyrec&ttyplay 기술을 사용하여 제작하였다. Ttyrec의 스크린 레코딩 기능은 터미널(terminal)의 툴 조작으로 기록 저장이 가능한데, 프로세싱(processing) 중에 ttyplay를 사용하여 이러한 기록들을 재생할 수 있다. ttyrec 와 ttyplay는 오픈소스(open source)여서 다운받을 수 있는 경로가 비교적 많기에 본 논문에서는 해당 코드를 담지 않았다.

아래는 최종 작품의 핵심 코드이다.

갑골문 : 새의 코드



www.xinlin.art - /2019/1

lxl.snu.ac.kr - /2019/1

Oraclebirds.pde

```
boolean recording=false;
PFont zsFont ;
TTYParser parser;
Terminal terminal;
int BEGIN_TIME;
float SPEED = 3;
float defaultTextSizeValue2 = 20f;
String fontFilePath2 = "HYChenTiJiaGuWen-2.ttf";
float maxrots = .4;
int totLines = 1000;
Line[] lines = new Line[totLines];

void setup() {


  size(594, 841,P3D);
  pixelDensity(2);
  zsFont = loadFont("Monaco-10.vlw");
  textMode(SCREEN);
  parser = new TTYParser("2rec");
  terminal = new Terminal(80, 24);
  for(int i = 0; i < totLines; i++)
   lines[i] = randomLine();
  camera(0, 0, (height<width?height:width), 0, 0, 0, 0, 1, 0);
  noStroke();
}

Line randomLine() {
  return new Line(randomPoint(),randomPoint());
}

float e = 100;
Point randomPoint() {
  return new Point(random(-e,e),random(-e,e),random(-e,e));
}

float ti = 0;

void draw() {

  background(#f2f2f2);
  translate(-245, -349);
  fill(90);
  textFont(zsFont,8.2);
```

```
  for (int i = 0; i < parser.frames.size();i++) {
  DataFrame fr = (DataFrame)parser.frames.get(i);
  fr.update();
  }
  text(terminal.to_String(), 12, 22);
  translate(245, 350);
  textFont(createFont(fontFilePath2, defaultTextSizeValue2, true));
  ti+=0.01;
  rotateX(-ti);
  rotateY(-ti/2);
  rotateZ(-ti/3);
  Point c = new Point(sin(ti)*width,cos(ti)*height,0);
  for(int i = 0; i < totLines; i++)
   nearestPoint(lines[i],c).render();
  if(recording){
   saveFrame("output/img_####.png");
  }
}
void keyPressed(){
  if(key == 'r'|| key =='R'){
  recording=!recording;
  }

}
float interp(float low, float high, float t) {
  return ((high - low) * t) + low;
}
Point nearestPoint(Line line, Point C) {
  Point A = line.a;
  Point B = line.b;
  float t1 = A.x*A.x;
  float t2 = A.y*A.y;
  float t3 = A.z*A.z;
  float t4 = B.x*A.x;
  float t7 = B.y*A.y;
  float t10 = B.z*A.z;
  float t13 = -t1-t2-t3+t4-B.x*C.x+A.x*C.x+t7-B.y*C.y+A.y*C.y+t10-B.z*C.z+A.z*C.z;
  float t14 = B.x*B.x;
  float t15 = B.y*B.y;
  float t16 = B.z*B.z;
  float t23 = -t13/(t14+t1+t15+t2+t16+t3-2.0*t4-2.0*t7-2.0*t10);
  return line.pointAt(t23);
}
```

Line.pde

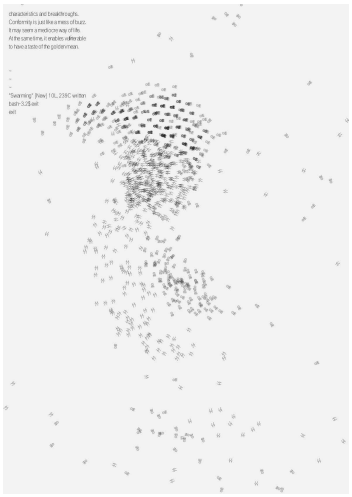
```
class Line {
  Point a, b;
  Line(Point a, Point b) {
    this.a = a;
    this.b = b;
  }
  Point pointAt(float t) {
    float x =  interp(a.x, b.x, t);
    float y =  interp(a.y, b.y, t);
    float z =  interp(a.z, b.z, t);
    return new Point(x, y, z);
  }
  void render() {
    line(a.x,a.y,a.z,b.x,b.y,b.z);
  }
}
```

Point.pde

```
class Point {
  float x, y, z;
  float rotx, roty, rotz;
  float rotsx, rotsy, rotsz;
  Point(float x, float y, float z) {
    this.x = x;
    this.y = y;
    this.z = z;
    rotx = 0;
    roty = 0;
    rotz = 0;
    rotsx = random(-maxrots,maxrots);
    rotsy = random(-maxrots,maxrots);
    rotsz = random(-maxrots,maxrots);
  }
  void render() {
    rotx += rotsx;
    roty += rotsy;
    rotz += rotsz;
    pushMatrix();
    translate(x,y,z);
    rotateX(rotx);
    rotateY(roty);
    fill(0);
    text('鳥',0,0,0);
    popMatrix();
  }
```

```
}
```

봉옹이지의 코드



www.xinlin.art - /2019/2

lxl.snu.ac.kr - /2019/2

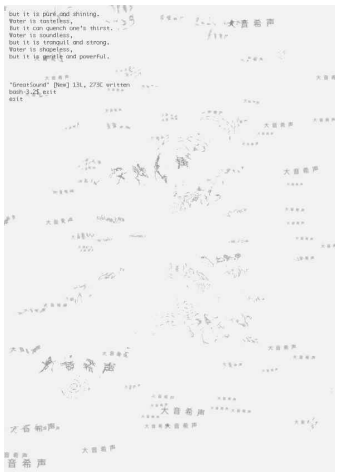
```
boolean recording =false;
TTYParser parser; Terminal terminal;
int BEGIN_TIME; float SPEED = 3;
float defaultTextSizeValue2 = 10f;
String fontFilePath2 = "HYChenTiJiaGuWen-2.ttf";
int fishNum = 500;
Fish[] fish = new Fish[fishNum];
boolean mouseDown = false;
void setup(){
  size(594,841); pixelDensity(1);
  background(255); fill(0);
  for(int i=0; i<fishNum; i++){
    fish[i] = new Fish(random(width), random(height));
  }

  textFont(loadFont("Monaco-10.vlw"));
  textMode(SCREEN);
  parser = new TTYParser("1rec");
  terminal = new Terminal(80, 24);
}
void mousePressed(){
  mouseDown = true;
}
void mouseReleased(){
  mouseDown = false;
}
void draw(){
 background(#f2f2f2);
  textFont(createFont(fontFilePath2, defaultTextSizeValue2, true));
  for(int i=0; i<fishNum; i++){
    fish[i].update();
  }
  for(int i = 0; i<fishNum-1; i++){
    for(int c = i+1; c<fishNum; c++){
      float d = dist(fish[i].x, fish[i].y, fish[c].x, fish[c].y)+1;
      if( d <= 15 ){
        fish[i].xv += .004*(fish[i].x - fish[c].x);
        fish[i].yv += .004*(fish[i].y - fish[c].y);
        fish[c].xv += .004*(fish[c].x - fish[i].x);
        fish[c].yv += .004*(fish[c].y - fish[i].y);
      }
```

```
      else if( d>50 && d <=80){
        fish[i].xv -= .000005*(fish[i].x - fish[c].x);
        fish[i].yv -= .000005*(fish[i].y - fish[c].y);
        fish[c].xv -= .000005*(fish[c].x - fish[i].x);
        fish[c].yv -= .000005*(fish[c].y - fish[i].y);
        fish[i].xv += ((fish[c].xv+fish[i].xv)/2-fish[i].xv)*.01;
        fish[i].yv += ((fish[c].yv+fish[i].yv)/2-fish[i].yv)*.01;
      }
    }
  }
  fill(0);
  for (int i = 0; i < parser.frames.size();i++) {
    DataFrame fr = (DataFrame)parser.frames.get(i);
    fr.update();
  }
  text(terminal.to_String(), 10, 20);
  if(recording){
  saveFrame("output/img_####.png");
  }
}
class Fish {
  float x,y;
  float xv = random(-1,1), yv = random(-1,1);
  float tailStep = 0, tailSpeed = random(2,3);
  float s = random(.05,.2);
  float sx = -90, sy = 0;
  float ax = -40, ay = 0;
  float bx = 5, by = 40;
  float cx = -40, cy = 0;
  float dx = 5, dy = -40;
  float ex = 10, ey = 0;
  float animOff = random(TWO_PI);
  Fish(float x, float y){
    this.x = x; this.y = y;
  }
  void update(){
    this.tailStep += this.tailSpeed;
    this.xv += (   noise(   this.x*.01 + PI,this.y*.01, millis()*.00002   )*2-1   )*.3;
    this.yv += (   noise(   this.x*.01 - PI,this.y*.01, millis()*.00002   )*2-1   )*.3;
    this.xv = constrain(this.xv, -2,2);
    this.yv = constrain(this.yv, -2,2);
    this.x += this.xv;
    this.y += this.yv;
    drawFish();
    if(this.x<-10){
      this.x = width+10;
    }
    else if(this.x>width+10){
      this.x = -10;
```

```
    }
    if(this.y<-10){
      this.y = height+10;
    }
    else if(this.y>height+10){
      this.y = -10;
    }
  }
  void drawFish(){
    sy = 30*sin( this.tailStep*.1 + this.animOff);
    pushMatrix();
    translate(this.x, this.y);
    rotate( atan2(this.yv, this.xv) );
    text('從',this.sx,this.sy,0);
    text('衆',this.ex,this.ex,this.ey);
    popMatrix();
  }
}
void keyPressed(){
  if(key == 'r'|| key =='R'){
  recording=!recording;
  }
}
```

대음희성 대상무형의 코드

dadaowuxing.pde



www.xinlin.art - /2019/3

lxl.snu.ac.kr - /2019/3

```
PFont zsFont;
TTYParser parser;
Terminal terminal;
int BEGIN_TIME;
float SPEED = 3;
int legNum=10;
int partNum=18;
Leg[][] leg = new Leg[legNum][partNum];
PVector[] Target = new PVector[legNum];
float[] speed = new float[legNum];
PFont font;
Food[] food = new Food[10000];



int foodNum = 190;
int foodCounter;
PGraphics ff;
```

```
Ripple ripple;

void setup() {
  size(594, 841,P3D);
  pixelDensity(1);
  zsFont = loadFont("Monaco-10.vlw");
  parser = new TTYParser("3rec");
  terminal = new Terminal(80, 24);
  ff= createGraphics(width,height,P3D);
  ff.smooth();
  ff.fill(255, 100);
  font = createFont("宋体",100,true);
  ff.textFont(font);
  ff.textAlign(CENTER);
  for (int i=0; i<foodNum; i++) {
    food[i] = new Food("大 音 希 聲",random(width),random(height),random(-500,500));
  }
  for (int i=0; i<legNum; i++) {
    float x = random(width);
    float y = random(height);
    for (int g=0; g<partNum;g++){
      float angle = g*TWO_PI/partNum;
      leg[i][g]= new Leg(x,y,300,i*TWO_PI/legNum,0.1+i*0.5, partNum, g);
      Target[i] = new PVector();
      speed[i] = 100;
    }
  }
  ripple = new Ripple();
  frameRate(30);
}

void draw() {
  pushMatrix();
  ff.beginDraw();
  ff.background(#f2f2f2);
  foodCounter=0;
  for (int i=0;i<food.length; i++) {
    if (food[i]!= null && !food[i].consumed) {
      food[i].display();
      foodCounter++;
    }
  }
  println("There is "+foodCounter+" pieces of food left.");
  for (int i=0; i<legNum; i++) {
    float randomNumber =random(10);
    if (randomNumber>9.8 && randomNumber <=9.9)
Target[i].set(random(width),random(height),random(-800,800));
    if (randomNumber>9.9) {
      int iWantFood = (int)random(foodNum);
```

```
        if (food[i]!= null && !food[i].consumed)
 Target[i].set(food[iWantFood].pos.x,food[iWantFood].pos.y,food[iWantFood].pos.z);
      }
      speed[i] +=0.05;
      for (int g=0; g<partNum;g++){
        leg[i][g].drawLeg(Target[i],speed[i]);
      }
      for (int j =0; j<foodNum; j++) {
        float d = PVector.dist(food[j].pos,leg[i][0].initia);
        if (d<50) {
          food[j].consumed =true;
          leg[i][0].life++;
          hit(food[j].pos.x,food[j].pos.y);
          if(speed[i]>21) speed[i] -=10;
        }
      }
    }
    ff.endDraw();
    loadPixels();
    ff.loadPixels();
    for (int loc = 0; loc < width * height; loc++) {
      pixels[loc] = ripple.col[loc];
    }
    updatePixels();
    ripple.newframe();
    hit(random(200,400),random(300,600));
    hit(random(width,height),random(width,height));
    hit(random(width,height),random(width,height));
    if(foodCounter<=100){ addfood();}
    popMatrix();
    fill(90);
    textFont(zsFont);
    for (int i = 0; i < parser.frames.size();i++) {
      DataFrame fr = (DataFrame)parser.frames.get(i);
      fr.update();
    }
    text(terminal.to_String(), 10, 20);
}

void addfood() {
foodNum++;
food[foodNum-1] =new Food("大 道 无 形", 100, 100,random(-500,500));
}
```

Food.pde

```
class Food {
  PVector pos,vel;
  float quantity,a,b,c,angle,BoWa,speed;
  String textFood;
  char[] letters;
  boolean consumed=false;
  int g,co;
  float d0=10;
  float d1=sqrt(sq(d0)+sq(3.3));
  float d2=sqrt(sq(2*d0)+sq(3.3));
  float d3=sqrt(sq(3*d0)+sq(2));
  float an1=atan2(3.3,d1);
  float an2=atan2(3.3,d2);
  float an3=atan2(2,d3);
  Food(String _text, float _x,float _y,float _z) {
    pos = new PVector(_x,_y,_z);
    textFood = _text;
    quantity =(float)textFood.length()/2;
    letters = textFood.toCharArray();
    pos = new PVector(random(width),random(height),random(-500,500));
    vel = new PVector();
    vel.normalize();
    co = int(random(20,70));
    a = random(0,1);
    b = random(1,10);
  }
  void display() {
   if (!consumed) {
    move();
    ff.pushMatrix();
    ff.noStroke();
    ff.fill(0,100);
    ff.translate(pos.x,pos.y,pos.z);
    rotateY(atan2(-vel.z,vel.x));
    rotateZ(acos(vel.y/vel.mag()));
    ff.textFont(font);
    ff.textAlign(CENTER);
    ff.textSize(10);
    drawBody();
    ff.popMatrix();
   }
  }
 void move() {
  a += .005;
  b += .035;
  if (random(10)>9.5)  {
  speed = 5*noise(b)/random(0.1,5);
```

```
  } else {
  speed = noise(b)*5;
  }
  vel.add(random(-0.05,0.05),random(-0.05,0.05),random(-0.05,0.05));
  vel.normalize();
  vel.mult(speed);
  pos.add(vel);
  if (pos.x>width+50) pos.x=-50;
  if (pos.x<-50) pos.x= width+50;
  if (pos.y>height+50) pos.y =-50;
  if (pos.y<-50) pos.y =height+50;
  if (pos.z>500) pos.z =-500;
  if (pos.z<-500) pos.z =500;
  c += 0.1*speed;
  BoWa=sin(c)*PI/15;
}
void drawBody() {
  for (g=0;g<letters.length; g++) {
    ff.translate(6,0);
    //ff.translate(textWidth(letters[g]),0);
    ff.rotate(BoWa/8);
    ff.textSize(8);
    ff.text(letters[g],0,0);
  }
 }
}
```

led.pde

```
class Leg {
  int jointsNum = 10;
  int counter,life;
  PVector[][] joints = new PVector[jointsNum][3];
  PVector initia,driver,target,ankle;
  float driverAngle,randomness,rotation,angle,O,Oangle;
  float speed =100;
  float[] armLength= new float[3];
  Leg(float xin, float yin, float zin, float _rotation, float _randomness, int _partNum, int _counter) {
    for (int i=0; i<jointsNum; i++) {
      for (int g= 0; g<3;g++) {
      joints[i][g] = new PVector(0,0,0);
      }
    }
    for (int g= 0; g<3;g++) {
      armLength[g] = random(9,12);
    }
    counter= _counter;
    angle = counter*TWO_PI/_partNum;
```

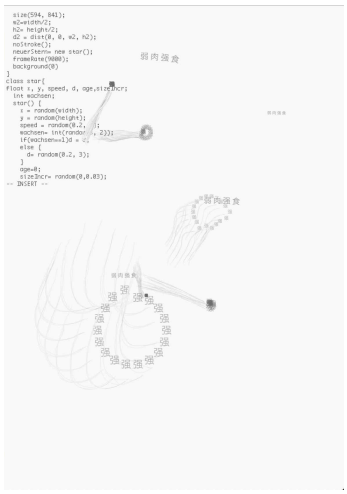
```
    initia = new PVector(xin,yin,zin);
    driver = new PVector(xin,yin,zin);
    target = new PVector(xin,yin,zin);
    rotation = _rotation;
    randomness = _randomness;
  }

  void drawLeg(PVector _target, float _speed) {
    target= _target;
    speed = _speed;
    driver.x += (target.x-driver.x)/speed;
    driver.y += (target.y-driver.y)/speed;
    driver.z += (target.z-driver.z)/speed;
    initia=driver.get();
    if (PVector.dist(driver,target)>30) {
    Oangle+=0.1;
    O=0.8*cos(Oangle)+0.2;
    driver.add(O*cos(angle),O*sin(angle),0);
    }
    for (int g= 0; g<3;g++) { joints[0][g]=driver.get(); }
    for (int g= 0; g<3;g++) {
    ff.noFill();
    ff.pushMatrix();
    ff.stroke(255,10);
    ff.point(initia.x, initia.y, initia.z);
    ff.stroke(255,40);
    ff.line(initia.x, initia.y, initia.z, joints[0][g].x,joints[0][g].y,joints[0][g].z);
    ff.popMatrix();
    ff.beginShape();
    ff.noFill();
    ff.stroke(255,10);
    ff.strokeWeight(2);
    ff.curveVertex(joints[0][g].x,joints[0][g].y,joints[0][g].z);
    for (int i=1; i<jointsNum; i++) {
      float angleZ = atan2(joints[i][g].y-joints[i-1][g].y,joints[i][g].x-joints[i-1][g].x);
      float angleY = atan2(joints[i][g].z-joints[i-1][g].z,joints[i][g].x-joints[i-1][g].x);
      float angleX = atan2(joints[i][g].y-joints[i-1][g].y,joints[i][g].z-joints[i-1][g].z);
      joints[i][g].x = joints[i-1][g].x + armLength[g]*cos(angleZ);
      joints[i][g].y = joints[i-1][g].y + armLength[g]*sin(angleZ);
      joints[i][g].z = joints[i-1][g].z + armLength[g]*cos(angleX);
      ff.curveVertex(joints[i][g].x+random(-i*0.3,i*0.3),joints[i][g].y+random(-i*0.3,i*0.3),joints[i][g].z);
    }
    ff.endShape();
  }
 }
}
```

Ripple.pde

```
class Ripple {
  int i, a, b;
  int oldind, newind, mapind;
  short ripplemap[];
  int col[];
  int riprad;
  int rwidth, rheight;
  int ttexture[];
  int ssize;

  Ripple() {
    riprad = 3;
    rwidth = width >> 1;
    rheight = height >> 1;
    ssize = width * (height + 2) * 2;
    ripplemap = new short[ssize];
    col = new int[width * height];
    ttexture = new int[width * height];
    oldind = width;
    newind = width * (height + 3);
  }

  void newframe() {
    i = oldind;
    oldind = newind;
    newind = i;
    i = 0;
    mapind = oldind;
    for (int y = 0; y < height; y++) {
      for (int x = 0; x < width; x++) {
      short data = (short)((ripplemap[mapind - width] + ripplemap[mapind + width] +
ripplemap[mapind - 1] + ripplemap[mapind + 1]) >> 1);
      data -= ripplemap[newind + i];
      data -= data >> 5;
       if (x == 0 || y == 0)
          ripplemap[newind + i] = 0;
        else
          ripplemap[newind + i] = data;
          data = (short)(1024 - data);
          a = ((x - rwidth) * data / 1024) + rwidth;
          b = ((y - rheight) * data / 1024) + rheight;
        if (a >= width)
          a = width - 1;
        if (a < 0)
          a = 0;
```

```
          if (b >= height)
            b = height-1;
          if (b < 0)
            b=0;
          col[i] = ff.pixels[a + (b * width)];
          mapind++;
          i++;
        }
      }
    }
}
void mouseDragged() {
hit(mouseX,mouseY);
foodNum++;
food[foodNum-1] =new Food("New food!!!", mouseX, mouseY,random(-500,500));
}
void hit(float _foodX, float _foodY) {
  int foodX =(int)_foodX;
  int foodY = (int)_foodY;
  for (int j = foodY - ripple.riprad; j < foodY + ripple.riprad; j++) {
    for (int k = foodX - ripple.riprad; k < foodX + ripple.riprad; k++) {
      if (j >= 0 && j < height && k>= 0 && k < width) {
        ripple.ripplemap[ripple.oldind + (j * width) + k] += 512;
      }
    }
  }
}
```

약육강식의 코드



www.xinlin.art - /2019/4

lxl.snu.ac.kr - /2019/4

ruorouqiangshi.pde

```
int legNum=6;
int partNum=18;
Leg[][] leg = new Leg[legNum][partNum];
PVector[] Target = new PVector[legNum];
float[] speed = new float[legNum];
PFont font;
Food[] food = new Food[1000];
int foodNum = 18;
int foodCounter;

PGraphics ff;
PFont zsFont;
TTYParser parser;
```

```
Terminal terminal;
int BEGIN_TIME;
float SPEED = 3;

void setup() {
  size(594, 841,P3D);
  pixelDensity(2);
  zsFont = loadFont("Monaco-10.vlw");
  parser = new TTYParser("3rec");
  terminal = new Terminal(80, 24);
  ff= createGraphics(width,height,P3D);
  ff.smooth();
  ff.fill(255, 100);
  font = createFont("宋体",100,true);
  ff.textFont(font);
  ff.textAlign(CENTER);
  for (int i=0; i<foodNum; i++) {
    food[i] = new Food("弱 肉 强 食",random(width),random(height),random(-500,500));
  }
  for (int i=0; i<legNum; i++) {
    float x = random(width);
    float y = random(height);
    for (int g=0; g<partNum;g++){
      float angle = g*TWO_PI/partNum;
      leg[i][g]= new Leg(x,y,300,i*TWO_PI/legNum,0.1+i*0.5, partNum, g);
      Target[i] = new PVector();
      speed[i] = 100;
    }
  }
  frameRate(30);
}

void draw() {
  pushMatrix();
  ff.beginDraw();
  ff.background(#f2f2f2);
  foodCounter=0;
  for (int i=0;i<food.length; i++) {
    if (food[i]!= null && !food[i].consumed) {
      food[i].display();
      foodCounter++;
    }
  }
  println("There is "+foodCounter+" pieces of food left.");
  for (int i=0; i<legNum; i++) {
    float randomNumber =random(10);
    if (randomNumber>9.8 && randomNumber <=9.9)
Target[i].set(random(width),random(height),random(-800,800));
    if (randomNumber>9.9) {
```

```
      int iWantFood = (int)random(foodNum);
      if (food[i]!= null && !food[i].consumed)
        Target[i].set(food[iWantFood].pos.x,food[iWantFood].pos.y,food[iWantFood].pos.z);
    }
    speed[i] +=0.05;
    for (int g=0; g<partNum;g++){
      leg[i][g].drawLeg(Target[i],speed[i]);
    }
    for (int j =0; j<foodNum; j++) {
      float d = PVector.dist(food[j].pos,leg[i][0].initia);
      if (d<50) {
        food[j].consumed =true;
        leg[i][0].life++;
        if(speed[i]>21) speed[i] -=10;
      }
    }
  }
 if(foodCounter<=8){
   addfood();
 }
  popMatrix();
  fill(90);
  textFont(zsFont);
  for (int i = 0; i < parser.frames.size();i++) {
    DataFrame fr = (DataFrame)parser.frames.get(i);
    fr.update();
  }
  text(terminal.to_String(), 10, 20);
}
void addfood() {
foodNum++;
food[foodNum-1] =new Food("适 者 生 存", 100, 100,random(-500,500));
}
```

food.pde

```
class Food {
  PVector pos,vel;
  float quantity,a,b,c,angle,BoWa,speed;
  String textFood;
  char[] letters;
  boolean consumed=false;
  int g,co;
  float d0=10;
  float d1=sqrt(sq(d0)+sq(3.3));
  float d2=sqrt(sq(2*d0)+sq(3.3));
  float d3=sqrt(sq(3*d0)+sq(2));
```

```
 float an1=atan2(3.3,d1);
 float an2=atan2(3.3,d2);
 float an3=atan2(2,d3);

 Food(String _text, float _x,float _y,float _z) {
   pos = new PVector(_x,_y,_z);
   textFood = _text;
   quantity =(float)textFood.length()/2;
   letters = textFood.toCharArray();
   pos = new PVector(random(width),random(height),random(-500,500));
   vel = new PVector();
   vel.normalize();
   co = int(random(20,70));
   a = random(0,1);
   b = random(1,10);
 }
 void display() {
  if (!consumed) {
   move();
   ff.pushMatrix();
   ff.noStroke();
   ff.fill(0,100);
   ff.translate(pos.x,pos.y,pos.z);
   rotateY(atan2(-vel.z,vel.x));
   rotateZ(acos(vel.y/vel.mag()));
   ff.textFont(font);
   ff.textAlign(CENTER);
   ff.textSize(10);
   drawBody();
   ff.popMatrix();
  }
 }
void move() {
 a += .005;
 b += .035;
 if (random(10)>9.5)  {
 speed = 5*noise(b)/random(0.1,5);
 } else {
 speed = noise(b)*5;
 }
 vel.add(random(-0.05,0.05),random(-0.05,0.05),random(-0.05,0.05));
 vel.normalize();
 vel.mult(speed);
 pos.add(vel);
 if (pos.x>width+50) pos.x=-50;
 if (pos.x<-50) pos.x= width+50;
 if (pos.y>height+50) pos.y =-50;
 if (pos.y<-50) pos.y =height+50;
 if (pos.z>500) pos.z =-500;
```

```
  if (pos.z<-500) pos.z =500;
  c += 0.1*speed;
  BoWa=sin(c)*PI/15;
}
void drawBody() {
  for (g=0;g<letters.length; g++) {
    ff.translate(6,0);
    ff.rotate(BoWa/8);
    ff.textSize(10);
    ff.text(letters[g],0,0);
  }
 }
}
```

leg.pde

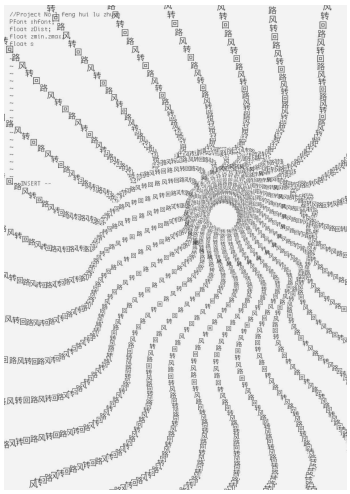
```
class Leg {
  int jointsNum = 12
  int counter,life;
  PVector[][] joints = new PVector[jointsNum][3];
  PVector initia,driver,target,ankle;
  float driverAngle,randomness,rotation,angle,O,Oangle;
  float speed =100;
  float[] armLength= new float[3];
  Leg(float xin, float yin, float zin, float _rotation, float _randomness, int _partNum, int _counter) {
    for (int i=0; i<jointsNum; i++) {
      for (int g= 0; g<3;g++) {
      joints[i][g] = new PVector(0,0,0);
      }
    }
    for (int g= 0; g<3;g++) {
      armLength[g] = random(9,12);
    }
    counter= _counter;
    angle = counter*TWO_PI/_partNum;
    initia = new PVector(xin,yin,zin);
    driver = new PVector(xin,yin,zin);
    target = new PVector(xin,yin,zin);
    rotation = _rotation;
    randomness = _randomness;
  }

  void drawLeg(PVector _target, float _speed) {
    target= _target;
    speed = _speed;
    driver.x += (target.x-driver.x)/speed;
    driver.y += (target.y-driver.y)/speed;
```

```
    driver.z += (target.z-driver.z)/speed;
    initia=driver.get();
    if (PVector.dist(driver,target)>30) {
    Oangle+=0.1;
    O=0.8*cos(Oangle)+0.2;
    driver.add(O*cos(angle),O*sin(angle),0);
    }
 for (int g= 0; g<3;g++) { joints[0][g]=driver.get(); }
 for (int g= 0; g<3;g++) {
    ff.noFill();
    ff.pushMatrix();
    ff.fill(100,50);
    String ttt ="强";
    ff.text(ttt,initia.x, initia.y, initia.z);
    ff.stroke(100,40);
    ff.line(initia.x, initia.y, initia.z, joints[0][g].x,joints[0][g].y,joints[0][g].z);
    ff.popMatrix();
    ff.beginShape();
    ff.noFill();
    ff.stroke(0,10);
    ff.strokeWeight(1);
    ff.curveVertex(joints[0][g].x,joints[0][g].y,joints[0][g].z);
    for (int i=1; i<jointsNum; i++) {
      float angleZ = atan2(joints[i][g].y-joints[i-1][g].y,joints[i][g].x-joints[i-1][g].x);
      float angleY = atan2(joints[i][g].z-joints[i-1][g].z,joints[i][g].x-joints[i-1][g].x);
      float angleX = atan2(joints[i][g].y-joints[i-1][g].y,joints[i][g].z-joints[i-1][g].z);
      joints[i][g].x = joints[i-1][g].x + armLength[g]*cos(angleZ);
      joints[i][g].y = joints[i-1][g].y + armLength[g]*sin(angleZ);
      joints[i][g].z = joints[i-1][g].z + armLength[g]*cos(angleX);
      ff.curveVertex(joints[i][g].x+random(-i*0.1,i*0.1),joints[i][g].y+random(-i*0.1,i*0.1),joints[i][g].z);
       }
    ff.endShape();
  }
 }
}
```

풍회로전의 코드

```
PFont zsFont,xhFont;
TTYParser parser;
Terminal terminal;
AbstractBackground currentBackground;
int BEGIN_TIME;
float SPEED = 3;
float zDist = 11, zmin = -150, zmax = 260, zstep = 0.8, rad = 630;
```

```
int nb = int((zmax - zmin) / zDist)*2;
PVector[] circles = new PVector[nb];
Boolean dots = true;

void setup() {
  size(594, 841,P3D);
  pixelDensity(2);
  currentBackground = new SandyBackground();
  zsFont = loadFont("Monaco-10.vlw");
  textMode(SCREEN);
  xhFont = createFont("xh W4 Light.TTF",10);
  parser = new TTYParser("1rec");
  terminal = new Terminal(80, 24);
  for (int i = 0; i < nb; i++) {
    circles[i] = new PVector(0, 0, map(i, 0, nb-1, zmax, zmin));
  }

}

void draw(){
  background(#f2f2f2);
  fill(90,228);
  textFont(zsFont);
  for (int i = 0; i < parser.frames.size();i++) {
    DataFrame fr = (DataFrame)parser.frames.get(i);
    fr.update();
  }
  text(terminal.to_String(), 10, 20);
  textFont(xhFont,10);
  fill(#363636);
  translate(width/2, height/2);
  PVector pv;
  float fc = (float)frameCount, a;
  if (dots) beginShape(POINTS);
  for (int i = 0; i < nb; i++) {
    pv = circles[i];
    pv.z += zstep;
    pv.x = (noise((fc*2 + pv.z) / 550) - .5) * 190 * map(pv.z, zmin, zmax, 11, 0);
    pv.y = (noise((fc*2 - 3000 - pv.z) / 550) - .5) * 200 * map(pv.z, zmin, zmax, 11, 0);
    a = map(pv.z, zmin, zmax, 0, 255);
    float r = map(pv.z, zmin, zmax, rad*.1, rad);
    if (dots) {
      float jmax = 30;
      String ttt = i%4==0?"風":(i%4==2?"回":(i%4==3?"路":"轉"));
      for (int j  = 0; j < jmax; j++)
      {
        text(ttt,pv.x + r*cos(j*TWO_PI/jmax + fc/40)/2, pv.y + r*sin(j*TWO_PI/jmax + fc/40)/2, pv.z);
      }
    } else {
```

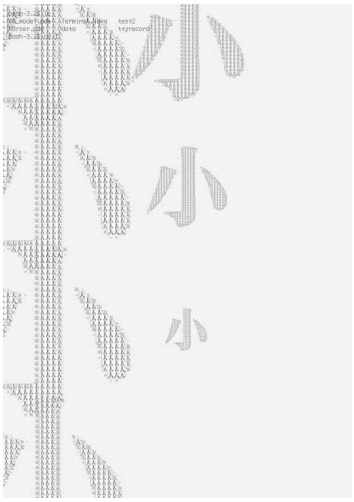


www.xinlin.art - /2019/5

lxl.snu.ac.kr - /2019/5

```
      pushMatrix();
      translate(pv.x, pv.y, pv.z);
      ellipse(0, 0, r, r);
      popMatrix();
    }
    if (pv.z > zmax) {
      circles[i].z = zmin;
    }
  }
  if (dots) endShape();
}

abstract class AbstractBackground {
  AbstractBackground() {
  }
  abstract void display();
}
final class SandyBackground extends AbstractBackground {
  final PGraphics graphics;

  SandyBackground(float hueParameter, float saturationParameter, float minBrightness, float maxBrightness, int
xSize, int ySize) {
    graphics = createGraphics(xSize, ySize);
    graphics.beginDraw();
    graphics.loadPixels();
    for (int x = 0; x < xSize; x++) {
      for (int y = 0; y < ySize; y++) {
        graphics.pixels[x + y * xSize] = color(hueParameter, saturationParameter, random(minBrightness,
maxBrightness));
      }
    }
    graphics.updatePixels();
    graphics.endDraw();
  }
  SandyBackground(float hueParameter, float saturationParameter, float minBrightness, float maxBrightness) {
    this(hueParameter, saturationParameter, minBrightness, maxBrightness, width, height);
  }
  SandyBackground() {
    this(0f, 0f, 95f, 100f);
  }

  void display() {
    image(graphics, 0f, 0f);
  }
}
```

큰 것, 작은 것의 코드



www.xinlin.art - /2019/6

lxl.snu.ac.kr - /2019/6

```
PFont zsFont;
PImage ap,bp;
PGraphics pg;
PGraphics pg1;
int stt = 100;
int [] pa = new int [961];
int [] sa = new int [961];
float ss1 = 0;
float ss2 = 0;
float gate = 0;
float sc = 0;
float stp = 18.2;
PGraphics dele;
TTYParser parser;
Terminal terminal;


int BEGIN_TIME;
float SPEED = 3;
void setup(){
  size(594, 841,P2D);
  zsFont = loadFont("Monaco-10.vlw");
  parser = new TTYParser("test2");
  terminal = new Terminal(80, 24);
  smooth(8);
  ap = loadImage("xiao2.png");
  bp = loadImage("da1.png");
  translate(-10,90);
  getPos();
  dimg();
  ss1 = map(pa[15*31+15],stt,255,20,0);
  ss2 = map(sa[15*31+15],stt,255,20,0);
  gate = 19220/ss1;
  sc = gate;
  dele = createGraphics(594, 841);
}
void draw(){
  width = 620;
  height = 620;
  background(#f2f2f2);
  pushMatrix();
  translate(width/2,height/2);
  scale(sc);
  tint(245,240);
}
  for(int i = 0;i < pa.length;i++){
    float ss = map(pa[i],stt,255,20,0);
```

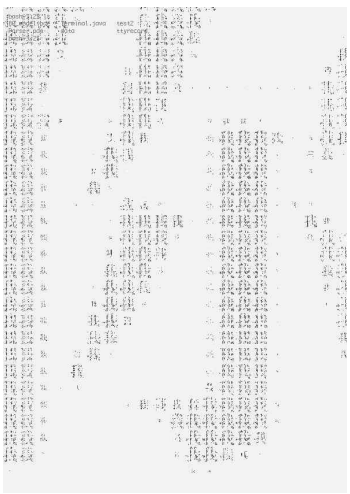
```
      int x = i%31;
      int y = i/31;
      if(x==15&&y==15){
        if(sc>32){
          continue;
        }
        else{
          tint(255,map(sc,1,32,255,0));
        }
      }
      image(pg1,(20*x+10)-ss/2-width/2,(20*y+10)-ss/2-height/2,ss,ss);
    }
    tint(255,255);
    scale((float)(ss1/620));
    if(sc<32){ tint(255,map(sc,1,32,0,255));}
    else{tint(255,map(sc,32,gate,255,100));}
    for(int i = 0;i < sa.length;i++){
      float ss = map(sa[i],stt,255,20,0);
      int x = i%31;
      int y = i/31;
      image(pg,(20*x+10)-ss/2-width/2,(20*y+10)-ss/2-height/2,ss,ss);
    }
    popMatrix();
    sc -= map(sc,gate,1,10,0.01);
    stp -= 0.001;
    if(sc<ss2/20){
      sc = gate;
      stp = 0.01;
    }
    fill(0,120);
    textFont(zsFont);
    for (int i = 0; i < parser.frames.size();i++) {
      DataFrame fr = (DataFrame)parser.frames.get(i);
      fr.update();
    }
    text(terminal.to_String(), 10, 20);
  }
  void del(){
    dele.beginDraw();
    dele.background(255);
    dele.pushMatrix();
    dele.translate(width/2,height/2);
    dele.scale(sc);
    dele.tint(255);
    for(int i = 0;i < pa.length;i++){
      float ss = map(pa[i],stt,255,20,0);
      int x = i%31;
      int y = i/31;
      if(x==15&&y==15){
```

```
        if(sc>32){
          continue;
        }
        else{
          dele.tint(255,map(sc,1,32,255,0));
        }
      }
      dele.image(pg1,(20*x+10)-ss/2-width/2,(20*y+10)-ss/2-height/2,ss,ss);
    }
    dele.tint(255,255);
    dele.scale((float)(ss1/620));
     if(sc<32){ dele.tint(255,map(sc,1,32,0,255));}
     else{ dele.tint(255,map(sc,32,gate,255,100));}
     for(int i = 0;i < sa.length;i++){
      float ss = map(sa[i],stt,255,20,0);
      int x = i%31;
      int y = i/31;
      dele.image(pg,(20*x+10)-ss/2-width/2,(20*y+10)-ss/2-height/2,ss,ss);
    }
    dele.popMatrix();
    sc -= map(sc,gate,1,10,0.01);
    stp -= 0.001;
    if(sc<ss2/20){
      sc = gate;
      stp = 0.01;
    }
    dele.endDraw();
}

void getPos(){
    PGraphics pg = createGraphics(31,31);
    pg.beginDraw();
    pg.image(ap,0,0,31,31);
    pg.endDraw();
    pg.loadPixels();
    for(int i = 0,ls = pg.pixels.length;i < ls;i++){
      color c = pg.pixels[i];
      pa[i] = floor(red(c));
    }
    PGraphics pg1 = createGraphics(31,31);
    pg1.beginDraw();
    pg1.image(bp,0,0,31,31);
    pg1.endDraw();
    pg1.loadPixels();
    for(int i = 0,ls = pg1.pixels.length;i < ls;i++){
      color c = pg1.pixels[i];
      sa[i] = floor(red(c));
    }
}
```

```
void dimg(){
  pg = createGraphics(620,620);
  pg.beginDraw();
  pg.pushMatrix();
  pg.translate(pg.width/2,pg.height/2);
  for(int i = 0;i < pa.length;i++){
    float ss = map(pa[i],stt,255,20,0);
    int x = i%31;
    int y = i/31;
    pg.image(bp,(20*x+10)-ss/2-pg.width/2,(20*y+10)-ss/2-pg.height/2,ss,ss);
  }
  pg.popMatrix();
  pg.endDraw();
  pg1 = createGraphics(620,620);
  pg1.beginDraw();
  pg1.pushMatrix();
  pg1.translate(pg1.width/2,pg1.height/2);
  for(int i = 0;i < sa.length;i++){
    float ss = map(sa[i],stt,255,20,0);
    int x = i%31;
    int y = i/31;
    pg1.image(ap,(20*x+10)-ss/2-pg1.width/2,(20*y+10)-ss/2-pg1.height/2,ss,ss);
  }
  pg1.popMatrix();
  pg1.endDraw();

}
```

네 , 내의 코드

```
PFont zsFont;
PImage ap,bp;
PGraphics pg;
PGraphics pg1;
int stt = 60;
TTYParser parser;
Terminal terminal;
int BEGIN_TIME;
float SPEED = 3;
void setup(){
  size(594, 841);
  zsFont = loadFont("Monaco-10.vlw");
  parser = new TTYParser("test2");
  terminal = new Terminal(80, 24);
  smooth(8);
  ap = loadImage("ni.png");
```



www.xinlin.art - /2019/7

lxl.snu.ac.kr - /2019/7

```
  bp = loadImage("wo.png");
  translate(-10,90);
  getPos();
  dimg();
}
float sc = 1;
float stp = 0.01;
void draw(){
  width = 620;
  height = 620;
  translate(-10,60);
  background(#f2f2f2);
  pushMatrix();
  translate(width/2,height/2);
  scale(sc);
  tint(255);
  if(sc>1&&sc<32){
  tint(255,map(sc,1,32,255,140));}
  for(int i = 0;i < pa.length;i++){
    float ss = map(pa[i],stt,255,20,0);
    int x = i%31;
    int y = i/31;
    if(x==15&&y==15){
      if(sc>32){
        continue;
      }
      else{
        tint(255,map(sc,1,32,255,0));
      }
    }
    image(pg1,(20*x+10)-ss/2-width/2,(20*y+10)-ss/2-height/2,ss,ss);
  }
  tint(255,255);
  float ss1 = map(pa[15*31+15],stt,255,20,0);
  float ss2 = map(sa[15*31+15],stt,255,20,0);
  float gate = 19220/ss1;
  scale((float)(ss1/620));
    if(sc<32){
      tint(255,map(sc,1,32,0,255));
    }
    else{
      tint(255,map(sc,32,gate,255,130));
    }
   for(int i = 0;i < sa.length;i++){
    float ss = map(sa[i],stt,255,20,0);
    int x = i%31;
    int y = i/31;
    image(pg,(20*x+10)-ss/2-width/2,(20*y+10)-ss/2-height/2,ss,ss);
   }
```

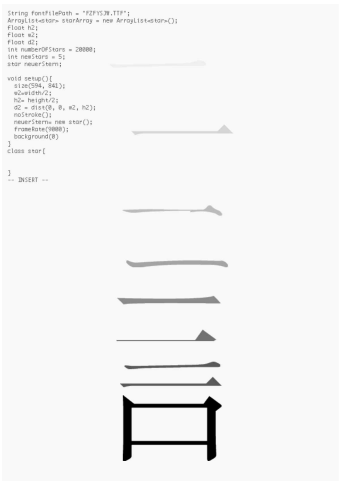
```
 }
  popMatrix();
  if(sc>1){
    sc+=stp;
    stp += map(sc,1,3.5,0,0.0005);
  }
  else{
    sc += map(sc,0.02,1,0.001,0.01);
  }
  if(sc>3.5){
    stp += 0.0005;
  }
  if(sc>gate){
    sc = (sc*ss2/(gate*20));
    stp = 0.01; }
  translate(10,-60);
  fill(120);
  textFont(zsFont);
  for (int i = 0; i < parser.frames.size();i++) {
    DataFrame fr = (DataFrame)parser.frames.get(i);
    fr.update(); }
  text(terminal.to_String(), 10, 20);
 }
  int [] pa = new int [961];
  int [] sa = new int [961];
  void getPos(){
  PGraphics pg = createGraphics(31,31);
  pg.beginDraw();
  pg.image(ap,0,0,31,31);
  pg.endDraw();
  pg.loadPixels();
  for(int i = 0,ls = pg.pixels.length;i < ls;i++){
    color c = pg.pixels[i];
    pa[i] = floor(red(c));
  }
  PGraphics pg1 = createGraphics(31,31);
  pg1.beginDraw();
  pg1.image(bp,0,0,31,31);
  pg1.endDraw();
  pg1.loadPixels();
  for(int i = 0,ls = pg1.pixels.length;i < ls;i++){
    color c = pg1.pixels[i];
    sa[i] = floor(red(c));
  }
}
void dimg(){
  pg = createGraphics(620,620);
  pg.beginDraw();
  pg.pushMatrix();
```

```
  pg.translate(pg.width/2,pg.height/2);
  for(int i = 0;i < pa.length;i++){
    float ss = map(pa[i],stt,255,20,0);
    int x = i%31;
    int y = i/31;
    pg.image(bp,(20*x+10)-ss/2-pg.width/2,(20*y+10)-ss/2-pg.height/2,ss,ss);
  }
  pg.popMatrix();
  pg.endDraw();
  pg1 = createGraphics(620,620);
  pg1.beginDraw();
  pg1.pushMatrix();
  pg1.translate(pg1.width/2,pg1.height/2);
  for(int i = 0;i < sa.length;i++){
    float ss = map(sa[i],stt,255,20,0);
    int x = i%31;
    int y = i/31;
    pg1.image(ap,(20*x+10)-ss/2-pg1.width/2,(20*y+10)-ss/2-pg1.height/2,ss,ss);
  }
  pg1.popMatrix();
  pg1.endDraw();
}
```

말은 실수를 낳는다의 코드

yan.pde

```
PFont zsFont;
TTYParser parser;
Terminal terminal;
int BEGIN_TIME;
float SPEED = 3;
PImage img1,img2,img3,img4,img5,img6,img7;
ParticleSystem ps;
int t=0;

void setup() {
  size(594, 841,P3D);
  pixelDensity(2);
  zsFont = loadFont("Monaco-10.vlw");
  parser = new TTYParser("3rec");

  terminal = new Terminal(80, 24);
```



www.xinlin.art - /2019/8

lxl.snu.ac.kr - /2019/8

```
  img1 = loadImage("3.png");
  img2 = loadImage("4.png");
  img3 = loadImage("5.png");
  img4 = loadImage("2.png");
  img5 = loadImage("1.png");
  img6 = loadImage("6.png");
  img7 = loadImage("kou5.png");
  ps = new ParticleSystem(0,new PVector(width/2,height-175),img1, img2, img3, img4,img5,img6);
}

void draw() {
  background(#f2f2f2);
  float dx = map(400,0,width,-0.01,-0.1);
  PVector wind = new PVector(0,dx);
  ps.applyForce(wind);
  ps.run();

  for (int i = 0; i < 1; i++) {
    if(t%16==0)
    ps.addParticle();
  }
  noTint();
  image(img7,width/2+3,height-95);
  t+=1;
  fill(90);
  textFont(zsFont);
  for (int i = 0; i < parser.frames.size();i++) {
    DataFrame fr = (DataFrame)parser.frames.get(i);
    fr.update();
  }
  text(terminal.to_String(), 10, 20);
}
```

Particle.pde

```
class Particle {
  PVector pos;
  PVector vel;
  PVector acc;
  float lifespan;
  PImage img;
  Particle(PVector l,PImage img_) {
    acc = new PVector(0,0);
    float vx = randomGaussian()*0.1;
```

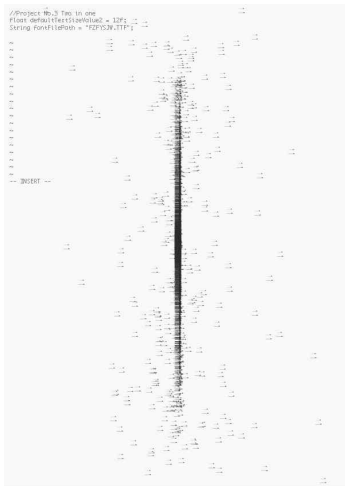
```
    float vy = randomGaussian()*0.3 - 1.0;
    vel = new PVector(vx,vy);
    pos = l.get();
    lifespan = 300.0;
    img = img_;
  }

  void run() {
    update();
    render();
  }

  void applyForce(PVector f) {
    acc.add(f);
  }

  void update() {
    vel.add(acc);
    pos.add(vel);
    lifespan -= 2.5;
    acc.mult(0);
  }

  void render() {
    imageMode(CENTER);
    tint(255,lifespan);
    image(img,pos.x,pos.y);
  }
  boolean isDead() {
    if (lifespan <= 0.0) {
      return true;
    } else {
      return false;
    }
  }
}
```

ParticleSystem.pde

```
class ParticleSystem {

  ArrayList<Particle> particles;
  PVector origin;
  PImage img1, img2, img3, img4,img5,img6;
  int img_count;
  ParticleSystem(int num, PVector v, PImage img_1, PImage img_2, PImage img_3, PImage img_4,PImage
```

```
img_5,PImage img_6) {
    particles = new ArrayList<Particle>();
    origin = v.get();
    img_count = 0;
    img1 = img_1;
    img2 = img_2;
    img3 = img_3;
    img4 = img_4;
    img5 = img_5;
    img6 = img_6;
    int i = 0;
    while(i < num) {
        if(img_count == 0) { particles.add(new Particle(origin, img1)); img_count++;}
        else if(img_count == 1) { particles.add(new Particle(origin, img2)); img_count++;}
        else if(img_count == 2) { particles.add(new Particle(origin, img3)); img_count++;}
        else if(img_count == 3) { particles.add(new Particle(origin, img4)); img_count++;}
        else if(img_count == 4) { particles.add(new Particle(origin, img5)); img_count++;}
        else if(img_count == 5) { particles.add(new Particle(origin, img6)); img_count = 0;}
        i++;
    }
  }

  void run() {
    for (int i = particles.size()-1; i >= 0; i--) {
      Particle p = particles.get(i);
      p.run();
      if (p.isDead()) {
        particles.remove(i);
      }
    }
  }

  void applyForce(PVector dir) {
    for (Particle p: particles) {
      p.applyForce(dir);
    }
  }

  void addParticle() {
     if(img_count == 0) { particles.add(new Particle(origin, img1)); img_count++;}
        else if(img_count == 1) { particles.add(new Particle(origin, img2)); img_count++;}
        else if(img_count == 2) { particles.add(new Particle(origin, img3)); img_count++;}
        else if(img_count == 3) { particles.add(new Particle(origin, img4)); img_count++;}
        else if(img_count == 4) { particles.add(new Particle(origin, img5)); img_count++;}
        else if(img_count == 5) { particles.add(new Particle(origin, img6)); img_count = 0;}
  }

}
```

둘이 모여 하나가 되다의 코드



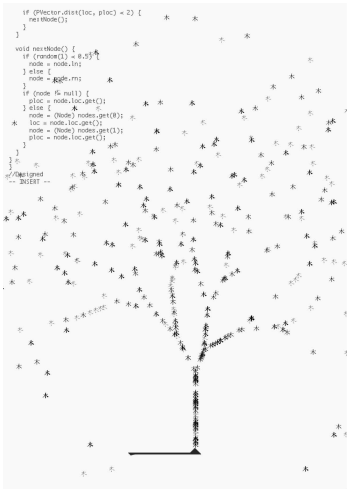
www.xinlin.art - /2019/9

lxl.snu.ac.kr - /2019/9

```
PFont zsFont;
float defaultTextSizeValue2 = 12f;
String fontFilePath2 = "FZFYSJW.TTF";
ArrayList<star> starArray = new ArrayList<star>();
float h2;//=height/2
float w2;//=width/2
float d2;//=diagonal/2
int numberOfStars = 20000;
int newStars =5;
star neuerStern;
TTYParser parser;
Terminal terminal;
int BEGIN_TIME;
float SPEED = 3;


void setup() {
  size(594, 841);
  zsFont = loadFont("Monaco-10.vlw");
  parser = new TTYParser("3rec");
  terminal = new Terminal(80, 24);
  w2=width/2;
  h2= height/2;
  d2 = dist(0, 0, w2, h2);
  noStroke();
  neuerStern= new star();
  frameRate(9000);
  background(0);
}
void draw() {
  background(#f2f2f2);
  fill(90);
  textFont(zsFont);
  for (int i = 0; i < parser.frames.size();i++) {
    DataFrame fr = (DataFrame)parser.frames.get(i);
    fr.update();
  }
  text(terminal.to_String(), 10, 20);
  fill(#f2f2f2, map(dist(600, 630, w2, h2), 0, d2, 255, -10));
  rect(0, 0, width, height);
  fill(#2b2a2a,random(100,220));
  textFont(createFont(fontFilePath2, defaultTextSizeValue2, true));
  for (int i = 0; i<newStars; i++) {
    starArray.add(new star());
  }
  for (int i = 0; i<starArray.size(); i++) {
```

```
    if(starArray.get(i).x<0||starArray.get(i).x>
width||starArray.get(i).y<0||starArray.get(i).y>height) starArray.remove(i);
    starArray.get(i).move();
    starArray.get(i).render();
  }
  if (starArray.size()>numberOfStars) {//
    for (int i = 0; i<newStars; i++) {
      starArray.remove(i);
    }
  }
}
class star {
  float x, y, speed, d, age,sizeIncr;
  int wachsen;
  star() {
    x = random(width);
    y = random(height);
    speed = random(0.2, 5);
    wachsen= int(random(0, 2));
    if(wachsen==1)d = 0;
    else {
      d= random(0.2, 3);
    }
    age=0;
    sizeIncr= random(0,0.03);
  }
  void render() {
   age++;
     if (age<200){
       if (wachsen==1){
         d+=sizeIncr;
         if (d>3||d<-3) d=3;
       }else {
         if (d>3||d<-3) d=3;
         d= d+0.2-0.6*noise(x, y, frameCount);
       }
     }
     else{
       if (d>3||d<-3) d=3;
     }
   text('二',x,y,d*(map(noise(x, y,0.001*frameCount),0,1,0.2,1.5)));
  }
  void move() {
    x =x-map(95, 0, width, -0.05*speed, 0.05*speed)*(w2-x);
    y =y-map(398, 0, height, -0.05*speed, 0.05*speed)*(h2-y);
  }
 }
}
```

나무 한 그루의 코드



www.xinlin.art - /2019/10

lxl.snu.ac.kr - /2019/10

```
TTYParser parser;
Terminal terminal;
PImage ben;
int BEGIN_TIME;
float SPEED = 3;
float defaultTextSizeValue = 11f;
String fontFilePath = "font_1_honokamin_subset.ttf";
ArrayList nodes = new ArrayList();
ArrayList nextNodes = new ArrayList();
Particle[] particles = new Particle[450];
int mainAngle = 30;
int amount = 40;
int distance = 130;
int endNodeAmount = 2;
```

```
void setup() {
  size(594, 841);
  pixelDensity(2);
  ben = loadImage("bensmall2.png");
  noStroke();
  fill(255);
  nodes.add(new Node(new PVector(width/2, height-50), 0, distance));
  Node n = new Node(new PVector(width/2, height-50-distance), 0, distance);
  n.addNextNodes();
  n.mn = (Node) nodes.get(0);
  nodes.add(n);
  amount = (int)pow(amount, 2);

  while (nodes.size() <= amount) {
    for (int i = 0; i < nextNodes.size(); i++) {
      Node n2 = (Node) nextNodes.get(i);
      nodes.add(n2);
    }
    nextNodes.clear();
    for (int i = 0; i < nodes.size(); i++) {
      if (i > nodes.size()-(endNodeAmount+1)) {
        Node n3 = (Node) nodes.get(i);
        if (nodes.size() <= amount) {
          n3.addNextNodes();
        }
      }
    }
    endNodeAmount *= 2;
  }
  for (int i = 0; i < particles.length; i++) {
```

```
    particles[i] = new Particle();
  }
  parser = new TTYParser("22rec");
  terminal = new Terminal(80, 24);
}
void draw() {
  background(#f2f2f2);
  translate(30, -20);
  image(ben, 186, 790);
  textFont(createFont(fontFilePath, defaultTextSizeValue, true));
  fill(0);
  for (int i = 0; i < particles.length; i++) {
    particles[i].draw();
    particles[i].move();
  }
  translate(-30, 20);
  textFont(loadFont("Monaco-10.vlw"));
  fill(90);
  for (int i = 0; i < parser.frames.size();i++) {
    DataFrame fr = (DataFrame)parser.frames.get(i);
    fr.update();
  }
  text(terminal.to_String(), 10, 20);
}
class Node {
  PVector loc;
  float angle, dist;
  Node ln, rn, mn;
  Node(PVector l, float a, float d) {
    loc = l.get();
    angle = a;
    dist = d-20;
  }
  void addNextNodes() {
    PVector vel = new PVector(sin(radians(angle-random(0, 15)+mainAngle))*(dist-random(20, 70)),
cos(radians(angle-random(0, 15)+mainAngle))*(dist-random(20, 70)));
    ln = new Node(new PVector(loc.x+vel.x, loc.y-vel.y), angle+mainAngle, distance);
    nextNodes.add(ln);
    vel = new PVector(sin(radians(angle+random(0, 15)-mainAngle))*(dist-random(20, 70)),
cos(radians(angle+random(0, 15)-mainAngle))*(dist-random(20, 70)));
    rn = new Node(new PVector(loc.x+vel.x, loc.y-vel.y), angle-mainAngle, distance);
    nextNodes.add(rn);
  }
}
class Particle {
  PVector loc, ploc;
  float speed = random(0.5, 1);
  Node node;
  Particle() {
```

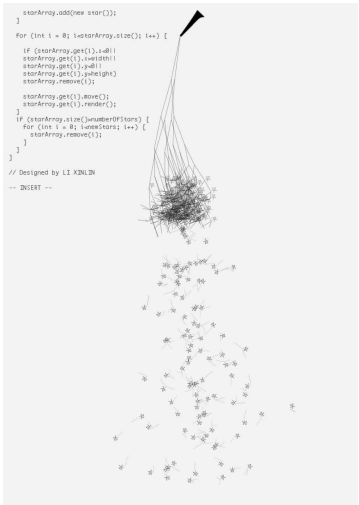
```
    Node ns = (Node) nodes.get(0);
    loc = ns.loc.get();
    node = (Node) nodes.get(1);
    ploc = node.loc.get();
  }
  void draw() {
     text('木',loc.x,loc.y,0);
  }
  void move() {
    PVector vel = new PVector(ploc.x - loc.x, ploc.y - loc.y);
    vel.normalize();
    vel.mult(speed);
    loc.add(vel);
    if (PVector.dist(loc, ploc) < 2) {
      nextNode();
    }
  }
  void nextNode() {
    if (random(1) < 0.5) {
      node = node.ln;
    } else {
      node = node.rn;
    }
    if (node != null) {
      ploc = node.loc.get();
    } else {
      node = (Node) nodes.get(0);
      loc = node.loc.get();
      node = (Node) nodes.get(1);
      ploc = node.loc.get();
    }
  }
}
```

풀 한 포기의 코드

caozhong.pde

```
PImage pie;
float defaultTextSizeValue = 11f;
float defaultTextSizeValue2 = 11f;
String fontFilePath = "FZShengSKSJW.TTF";
String fontFilePath2 = "FZShengSKSJW.TTF";
int t = 0;
ArrayList<drop> da = new ArrayList<drop>();
```

```
PFont zsFont;
TTYParser parser;
Terminal terminal;
int BEGIN_TIME;
float SPEED = 3;

void setup() {
  size(594, 841,P3D);
  pixelDensity(2);
  zsFont = loadFont("Monaco-10.vlw");
  pie = loadImage("pie.png");
  parser = new TTYParser("3rec");
  terminal = new Terminal(80, 24);
}

void draw() {
  background(#f2f2f2);


  image(pie, 296, 12);
  pushMatrix();
  translate(width / 2, 43);
  scale(0.9, 0.9);
  textFont(createFont(fontFilePath2, defaultTextSizeValue2, true));
  drawLine(0, 10, 30);
  for(int i = da.size()-1;i>=0;i--){
    drop d = da.get(i);
    d.draw();
    if(d.pos.y>height){
      da.remove(d);
    }
  }
  popMatrix();
  t += 1;
  fill(90);
  textFont(zsFont);
  for (int i = 0; i < parser.frames.size();i++) {
    DataFrame fr = (DataFrame)parser.frames.get(i);
    fr.update();
  }
  text(terminal.to_String(), 10, 20);
}
void drawLine(float x, float y, float r) {
  float nextX1 = x + 20 * sin(radians(r * 13 + t));
  float nextX2 = x - 20 * cos(radians(r * 13 + t));
  float nextY = y + r * 2.8;

  noStroke();
  fill(90);
  if(y<360&&y>280) {
```



www.xinlin.art - /2019/11

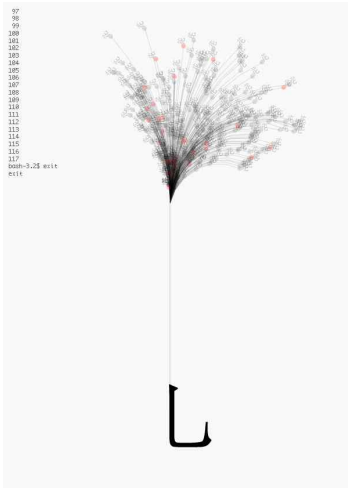
lxl.snu.ac.kr - /2019/11

```
      text('루',x,y,0);
      if(random(1)>0.999&&da.size()<300){
       da.add(new drop(new PVector(x,y)));
      }
    }
    stroke(25, 80);
    strokeWeight(1);

    if(y < 360){
      line(x, y, nextX1, nextY);
      line(x, y, nextX2, nextY);
    }
    if (r > 0) {
      drawLine(nextX1, nextY, r - 3.0);
      drawLine(nextX2, nextY, r - 5.5);
    }
}
```

drop.pde

```
class drop{
   PVector pos; float spy; float spx;
   float pr = random(30,300);
   float sd = random(9999);
   float rt = random(PI);
   float rts;
   PVector [] ds = new PVector[2];
   drop(PVector pos){
    this.pos = pos;
    spy = random(0.8,1);
    rts = map(spy,0.8,1,0.01,0.05);
    spx =(random(1)>0.5?1:-1)*random(0.1,0.3);
    if(true){
        float ls = random(5,20);
        float a = random(PI);
        ds[0] = new PVector(ls*cos(a),ls*sin(a));
        if(random(1)>0.5){
            ds[1] = new PVector(0,0);
        }
        else{
          float a1 = a + floor(random(6))*PI/6;
          ds[1] = new PVector(ls*cos(a1),ls*sin(a1));
        }
    }
 }
 void draw(){
    pushMatrix(); translate(pos.x,pos.y);
    rotate(rt);noStroke();fill(25);
```

```
    text('早',0,0,0);stroke(25,30);
    line(0,0,ds[0].x,ds[0].y); line(0,0,ds[1].x,ds[1].y);
    popMatrix();
    pos.x += 1.5*(0.5-noise(frameCount/pr+sd));
    pos.y += spy;
    rt += rts;
  }
}
```



www.xinlin.art - /2019/12
lxl.snu.ac.kr - /2019/12

꽃 한 떨기의 코드

yihua.pde

```
PImage gou;
PFont zsFont;
TTYParser parser;
Terminal terminal;
int BEGIN_TIME;

float SPEED = 3;
Limb[] limb = new Limb[300];
float ossilation=0;
float ossilationInc=1;
float t=0;
float stp = 0.03;
float defaultTextSizeValue = 15f;
String fontFilePath = "FZCuJinLJW.TTF";

void setup() {
  size(594, 841);
  pixelDensity(1);
  gou = loadImage("gou2.png");
  for (int i=0;i<limb.length;i++) {
    limb[i]= new Limb( int(random(500, 500) ), 300, (int)random(1, 7) );
  }
  zsFont = loadFont("Monaco-10.vlw");
  parser = new TTYParser("3rec");
  terminal = new Terminal(80, 24);
}

void draw() {
  textFont(createFont(fontFilePath, defaultTextSizeValue, true));
  pushMatrix();
  translate(-49,850);
  rotate(300/-200.0);
  ossilation+=ossilationInc;
```

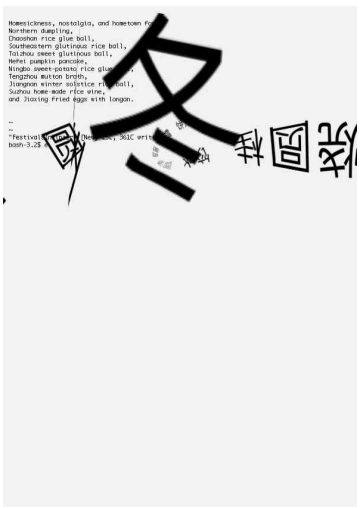
```
  background(#f2f2f2);
  for (int i=0;i<limb.length;i++) {
    limb[i].render();
  }
  if(t>100){
    stp*=-0.01;
  }
  if(t<0){
    stp*=-0.01;
  }
  t += stp;
  popMatrix();
  stroke(126,80);
  strokeWeight(1.5);
  line(287,340,287,700);
  image(gou, 286, 660);
  fill(90);
  textFont(zsFont);
  for (int i = 0; i < parser.frames.size();i++) {
    DataFrame fr = (DataFrame)parser.frames.get(i);
    fr.update();
  }
  text(terminal.to_String(), 10, 20);
}
```

Limb.pde

```
class Limb {
  int x=0;
  int y=0;
  int links=1;
  float len=50;
  int a=0;
  int d=0;

  public Limb(int x, int y, int links) {
    this.x=x;
    this.y=y;
    this.links=links;
    len=random(25, 50);
    a=(int)random(-25, 125);
    if(random(100) <50){ d=1; }else{ d=1;}
  }

  public void render() {
    pushMatrix();
    fill(0, 0, 0, 30);
```

```
    translate(x, y);
    for (int i=0;i<links;i++) {
      translate(len, 0);
      rotate( -sin(radians(a+(ossilation*d)/2+t )*t/100) /2);
      rotate( -cos(radians(a+(ossilation*d)*i/2+t )) /2);
      rect(0, 0, len, 1);
    }
    fill(0,46);
    text('花',len,0,10);
    noStroke();
    if (random(100) >90) {
      fill(255, 0, 0, 60);
    }
    else {
      fill(0, 0, 0, 30);
    }
    ellipse(len, 0, 8, 8);
    popMatrix();
  }
}
```

동절의 코드



www.xinlin.art - /2019/13

lxl.snu.ac.kr - /2019/13

zhi.pde

```
TTYParser parser;
Terminal terminal;
AbstractBackground currentBackground;
int BEGIN_TIME;
float SPEED = 3;
boolean showPath = true;
String path = "126,266%126...";
ArrayList chain = new ArrayList();
color c;
PFont font;
String tex[], plain, reversed;
int num = 51;
float count = 0;


int globalStep = 15;
float rtc = 0;
String reverseWords(String input[]) {
  String in = "";
  for (int i =0; i < input.length; i++) in += input[i]+" ";
  String tmp[] = splitTokens(in, " ");
  String out = "";
```

```
  for (int i = 0; i<tmp.length; i++) {
    out += new StringBuffer(tmp[i]).reverse().toString() +" ";
  }
  out = new StringBuffer(out).reverse().toString();
  return out;
}
ArrayList<PVector> pa = new ArrayList<PVector>();
PVector outVector = new PVector(0,0);
int pasize = 0;
void setup() {
  size(594, 841,P3D);
  showPath =! showPath;
  pixelDensity(1);
  currentBackground = new SandyBackground();
  c = color(random(255), random(255), random(255));
  tex = loadStrings("text.txt");
  reversed = reverseWords(tex);
  font = createFont("xh W4 Light.TTF", 40);
  String [] sa = path.split("%");
  for (int i = 0; i < sa.length; i++) {
    String [] pta = sa[i].split(",");
    pa.add(new PVector(Integer.parseInt(pta[0]), Integer.parseInt(pta[1])));
  }
  for (int i  =0; i<num; i++)
  chain.add(new Chain(i));
  noFill();
  background(0);

  outVector.x = pa.get(1750).x-pa.get(1700).x;
  outVector.y = pa.get(1750).y-pa.get(1700).y;
  outVector.normalize().mult(1);
  pasize = pa.size();
  colorMode(HSB, 255);
  textFont(loadFont("Monaco-10.vlw"));
  textMode(SCREEN);
  parser = new TTYParser("test2");
  terminal = new Terminal(80, 24);

}
void draw(){
  hint(DISABLE_DEPTH_TEST);
  background(#f2f2f2);
  if (showPath) {
    stroke(0);
    beginShape();
    for (int i = 0, ls = pa.size(); i < ls; i++) {
      PVector p = pa.get(i);
      vertex(p.x, p.y);
    }
```

```
    endShape();
  }
  noStroke();
  noFill();
  int idx = floor(count);
  float addC = (1+5*noise(frameCount/50));
  for (int i =0; i < chain.size(); i++) {
    Chain c  = (Chain)chain.get(i);
    if(count + addC<pasize){
      c.update(pa.get(idx), idx);
    }
    else{
      c.update(new PVector(c.pos.x+outVector.x,c.pos.y+outVector.y),idx);
    }
  }
  for (int i =chain.size()-1; i > 0; i--) {
    Chain c  = (Chain)chain.get(i);
    c.draw();
  }
  count += addC;
  if(count>pasize+num*globalStep){
    count = 0;
    for(int i = 0;i < chain.size();i++){
       Chain c = (Chain)chain.get(i);
       c.pos = new PVector(-90,height/2);
    }
  }
  fill(0);
  for (int i = 0; i < parser.frames.size();i++) {
    DataFrame fr = (DataFrame)parser.frames.get(i);
    fr.update();
  }
  text(terminal.to_String(), 10, 20);
}
void shapeIt() {
    for (int z = 10; z<30; z+=5) {
      stroke(0, 30);strokeWeight(z);
      noFill();beginShape();
      Chain c  = (Chain)chain.get(0);
      vertex(c.x1, c.y1);
    for (int i =0; i < chain.size(); i++) {
      c  = (Chain)chain.get(i);
      curveVertex(c.x1, c.y1);
    }
    endShape();
    beginShape();
    c  = (Chain)chain.get(chain.size()-1);
    vertex(c.x2, c.y2);
    for (int i = chain.size()-1; i > 0; i--) {
```

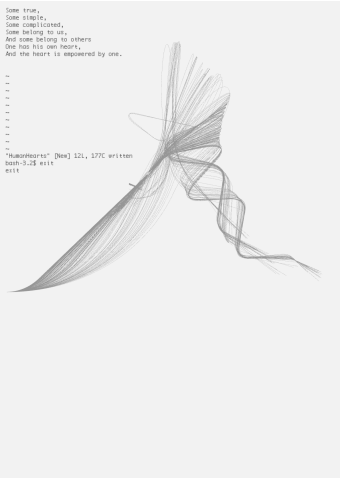
```
      c  = (Chain)chain.get(i);
      curveVertex(c.x2, c.y2);
    }
    c  = (Chain)chain.get(chain.size()-1);
    vertex(c.x2, c.y2);
    endShape();
  }
}
abstract class AbstractBackground {
  AbstractBackground() {
  }
  abstract void display();
}
final class SandyBackground extends AbstractBackground {
  final PGraphics graphics;
  SandyBackground(float hueParameter, float saturationParameter,
  float minBrightness, float maxBrightness, int xSize, int ySize) {
    graphics = createGraphics(xSize, ySize);
    graphics.beginDraw();
    graphics.loadPixels();
    for (int x = 0; x < xSize; x++) {
      for (int y = 0; y < ySize; y++) {
        graphics.pixels[x + y * xSize] =
        color(hueParameter, saturationParameter,
        random(minBrightness, maxBrightness));
      }
    }
    graphics.updatePixels();
    graphics.endDraw();
  }
  SandyBackground(float hueParameter, float saturationParameter,
  float minBrightness, float maxBrightness) {
    this(hueParameter, saturationParameter, minBrightness,
    maxBrightness, width, height);
  }
  SandyBackground() {
    this(0f, 0f, 95f, 100f);
  }
  void display() {
    image(graphics, 0f, 0f);
  }
}
```

Chain.pde

```
class Chain {
  PVector pos;
  Chain n;
  int id;
  float d, theta;
  float x1, x2, y1, y2;
  PGraphics mapa;
  float nsd;
  float seed = random(10000);
  int rd = floor(random(1000));
  int rd1 = floor(random(0,5));
  Chain(int _id) {
    id = _id;
    nsd = 0;
    if (id==0)
      n = this;
    else
      n = (Chain)chain.get(id-1);
    mapa = createGraphics(120, 120, JAVA2D);
    mapa.beginDraw();
    mapa.pushMatrix();
    mapa.translate(mapa.width/2,mapa.height/2);
    mapa.rotate(PI);
    mapa.textFont(font, 110);
    mapa.fill(0);
    mapa.textAlign(CENTER);
    mapa.text(reversed.charAt(reversed.length()-1-(id%reversed.length())),
    0, mapa.height/2-30);
    mapa.scale(-1, 1);
    mapa.popMatrix();
    mapa.endDraw();
    theta = 0;
    pos = new PVector(-90, height/2);
  }

  void update(PVector p,int idx) {
    int dis = chain.size()-1-this.id;
    int ls = dis*globalStep;
    int soutv = floor(this.rd1*noise(this.rd+frameCount/1000));
  }
    int idxpa = idx - ls-soutv;
    int idxoff = idxpa;
    if (id==pa.size()-1) {
      pos.x += (p.x-pos.x)/1.1;
      pos.y += (p.y-pos.y)/1.1;
    } else {
      if(idxpa<pasize){
```

```
        pos.x += (n.pos.x-pos.x)/(10.1);
        pos.y += (n.pos.y-pos.y)/(10.1);
        if(idxoff>=0&&idxpa<=pasize){
          PVector offset = pa.get(idxpa);
          PVector ori = pos.copy();
          PVector des = offset.copy();
          pos.add(des.sub(ori).mult(0.09));
        }
      }
      else{
        pos.x = p.x;
        pos.y = p.y;
      }
      d = dist(pos.x, pos.y, n.pos.x, n.pos.y);
      theta = atan2(n.pos.y-pos.y, n.pos.x-pos.x);
    }
    pushMatrix();
    translate((pos.x+n.pos.x)*.5, (pos.y+n.pos.y)*0.5);
    x1 = screenX(cos(theta+QUARTER_PI)*d, sin(theta+QUARTER_PI)*d);
    y1 = screenY(cos(theta+QUARTER_PI)*d, sin(theta+QUARTER_PI)*d);
    x2 = screenX(cos(theta-QUARTER_PI)*d, sin(theta-QUARTER_PI)*d);
    y2 = screenY(cos(theta-QUARTER_PI)*d, sin(theta-QUARTER_PI)*d);
    popMatrix();
  }

  void draw() {
    pushMatrix();
    translate(width/2,height/2);
    rotate(PI/26*(sin(rtc)));
    fill(0, 0);
    noStroke();
    beginShape();
    float tt = 100*map(d,0,40,0,0.5);
    float xo = width/2;
    float yo = height/2;
    tint(255, tt*20*map(this.rd1,0,5,0.5,1));
    texture(mapa);
    vertex(x2-xo, y2-yo, 0, 0);
    vertex(n.x2-xo, n.y2-yo, mapa.width, 0);
    vertex(n.x1-xo, n.y1-yo, mapa.width, mapa.height);
    vertex(x1-xo, y1-yo, 0, mapa.height);
    endShape(CLOSE);
    popMatrix();
    rtc += 0.0005;
  }
}
```

인간의 마음의 코드



www.xinlin.art - /2019/14

lxl.snu.ac.kr - /2019/14

```
import processing.svg.*;
TTYParser parser;
Terminal terminal;
int BEGIN_TIME;
float SPEED = 3;
boolean record;
boolean showPath = true;
Path[] p;
Vehicle[][] vehicles;
Vehicle v1;
Vehicle v2;
int seed=2;
PFont f;
PShape sap, sbp;
void init(){
  String[] svgList={"xin16.svg","xin16.svg","ren3.svg"};


  if(seed==svgList.length)seed=0;
  sap=loadShape(svgList[seed]);
  newPath();
}
void setup() {
  size(594, 841,P3D);
  pixelDensity(2);
  frameRate(10);
  textFont(loadFont("Monaco-10.vlw"));
  parser = new TTYParser("test2");
  terminal = new Terminal(80, 24);init();
  vehicles = new Vehicle[sap.getChildCount()][30];
  newPath();
  for (int num=0; num<sap.getChildCount(); num++) {
    Vehicle[] veh = new Vehicle[230];
    for (int i = 0; i < veh.length; i++) {
      veh[i] = new Vehicle(new PVector(30, 460), 2, random(0.01, 0.05));
    }
    vehicles[num]=veh;
  }
}
void draw(){
 if (record) { beginRecord(SVG, "frame1.svg");
}
 background(#f2f2f2);
 pushMatrix();
 translate(-20,50);
   for (int num=0; num<sap.getChildCount(); num++) {
    p[num].display();
```

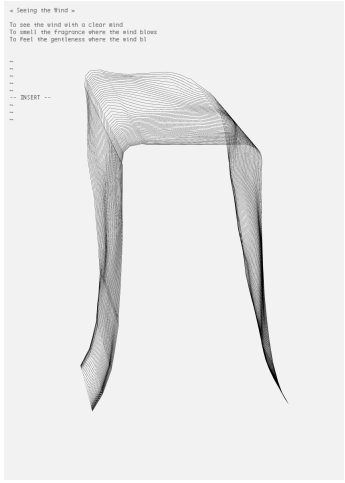
```
      for (int i = 0; i < vehicles[num].length; i++) {
        vehicles[num][i].follow(p[num]);
        vehicles[num][i].run(p[num]);
        vehicles[num][i].borders(p[num]);
      }
    }
      if(frameCount%660==659){
      init();
      seed++;
    }
  popMatrix();
  fill(90);

    for (int i = 0; i < parser.frames.size();i++) {
      DataFrame fr = (DataFrame)parser.frames.get(i);
      fr.update();
    }

    text(terminal.to_String(), 10, 20);
        if (record) {
      endRaw();
      record = false;
    }
}
void newPath() {
    p = new Path[0];
    for (int num=0; num<sap.getChildCount(); num++) {
      Path pp=new Path();
      sbp=sap.getChild(num);
      for (int i=0; i<sbp.getVertexCount(); i++) {
        pp.addPoint(sbp.getVertex(i).x, sbp.getVertex(i).y);
      }
      p=(Path[])append(p, pp);
    }
}

void keyPressed() {
    if (key == ' ') {
      showPath =! showPath;
    }
     if (key == 'r') {
      record = true;
    }
}

class Path {
    ArrayList<PVector> points;
    float radius;
    Path() {
```

```
    radius = 20;
    points = new ArrayList<PVector>();
  }
  void addPoint(float x, float y) {
    PVector point = new PVector(x, y);
    points.add(point);
  }
  PVector getStart() {
    return points.get(0);
  }
  PVector getEnd() {
    return points.get(points.size()-1);
  }

  void display() {
    if (showPath) {
      stroke(230);
      strokeWeight(radius*2);
      noFill();
      beginShape();
      for (PVector v : points) {
        vertex(v.x, v.y);
      }
      endShape();
    }
  }
}
class Vehicle {
  ArrayList<PVector> history = new ArrayList<PVector>();
  PVector loc;
  PVector vel;
  PVector acc;
  float r;
  float maxspeed;
  float maxforce;

  Vehicle(PVector l, float ms, float mf) {
    loc = l.get();
    r = 4.0;
    maxspeed = ms;
    maxforce = mf;
    acc = new PVector(0, 0);
    vel = new PVector(maxspeed, 0);
  }

  public void run(Path p) {
    update();
    display(p);
  }
```

```
void follow(Path p) {
  PVector predict = vel.get();
  predict.normalize();
  predict.mult(5);
  PVector predictLoc = PVector.add(loc, predict);

  PVector normal = null;
  PVector target = null;
  float worldRecord = 1000000;
  for (int i = 0; i < p.points.size()-1; i++) {
    PVector a = p.points.get(i);
    PVector b = p.points.get(i+1);
    PVector normalPoint = getNormalPoint(predictLoc, a, b);

    if (normalPoint.x < a.x || normalPoint.x > b.x) {
      normalPoint = b.get();
    }

    float distance = PVector.dist(predictLoc, normalPoint);
    if (distance < worldRecord) {
      worldRecord = distance;
      normal = normalPoint;

      PVector dir = PVector.sub(b, a);
      dir.normalize();
      dir.mult(10);

      target = normalPoint.get();
      target.add(dir);
    }
  }
  if (worldRecord > p.radius) {
    seek(target);
  }
}

PVector getNormalPoint(PVector p, PVector a, PVector b) {
  PVector ap = PVector.sub(p, a);
  PVector ab = PVector.sub(b, a);
  ab.normalize();
  ab.mult(ap.dot(ab));
  PVector normalPoint = PVector.add(a, ab);
  return normalPoint;
}

void update() {
  vel.add(acc);
  vel.limit(maxspeed);
  loc.add(vel);
```

```
    acc.mult(0);

    history.add(loc.get());
    if (history.size() > 5000) {
      history.remove(0);
    }
  }

  void applyForce(PVector f) {
    acc.add(f);
  }

  void seek(PVector target) {
    PVector desired = PVector.sub(target, loc);
    if (desired.mag() == 0) return;
    desired.normalize();
    desired.mult(maxspeed);
    PVector steer = PVector.sub(desired, vel);
    steer.limit(maxforce);
    applyForce(steer);
  }

  void display(Path p) {
    beginShape();
    strokeWeight(0.2);
    stroke(150);
    noFill();
    for (int i = 0; i < history.size(); i++) {
      PVector v = new PVector(history.get(i).x, history.get(i).y);
      vertex(v.x, v.y);
      if (history.get(i).x > width-1 ||history.get(i).x >
      p.getEnd().x || history.get(i).y > height-1) {
        endShape();
        history.clear();
      } else if (history.get(i).x < 1 || history.get(i).y < 1) {
        beginShape();
      }
    }
    endShape();
  }

  void borders(Path p) {
    if (loc.x > p.getEnd().x + r) {
      loc.x = p.getStart().x - r;
      loc.y = p.getStart().y + (loc.y - p.getEnd().y);
    }
  }
}
```

바람을 바라보다의 코드



www.xinlin.art - /2019/15

lxl.snu.ac.kr - /2019/15

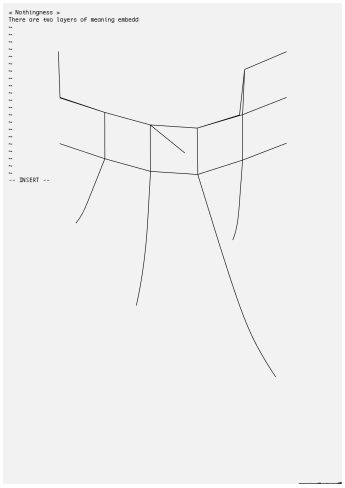
```
import processing.pdf.*;
import processing.svg.*;
TTYParser parser;
Terminal terminal;
AbstractBackground currentBackground;
int BEGIN_TIME;
float SPEED = 3;
boolean record;
NoisySquare[] noise_sqrs=new NoisySquare[0];
PShape sap, sbp;

void setup() {
  size(594, 841, P3D);
  pixelDensity(2);
  textFont(loadFont("Monaco-10.vlw"));


  textMode(SCREEN);
  parser = new TTYParser("test2");
  terminal = new Terminal(80, 24);
  strokeWeight(0.2);
  stroke(0);
  smooth();
  sap=loadShape("test15.svg");
  sbp=sap.getChild(0);
  for (int i=0; i<sbp.getVertexCount(); i++) {
    println(sbp.getVertex(i));
  }
  for (int i = 0; i < 50; i++) {
    float disp = norm(i, 0, 50)*200;
    float off = i*0.01;
    noise_sqrs=(NoisySquare[])
    append(noise_sqrs, new NoisySquare( disp, off));
  }
}
void draw() {
  if (record) {
    beginRaw(PDF, "3D.pdf");
  }
  background(#f2f2f2);
  fill(90);
  for (int i = 0; i < parser.frames.size(); i++) {
    DataFrame fr = (DataFrame)parser.frames.get(i);
    fr.update();
  }
  text(terminal.to_String(), 10, 20);
  noFill();
```

```
  scale(1.5);
  translate(-100, -300);
  for (int i = 0; i < noise_sqrs.length; i++) {
    noise_sqrs[i].run();
  }
  if (record) {
    endRaw();
    record = false;
  }
}
class Particle {
  PVector p_0, position, p_a, p_b;
  float noise_offset, noise_scale;
  Particle(
  float x_,
  float y_,
  float displace_amt_,
  float offset_) {
    p_0 = new PVector(x_, y_);
    position = p_0.copy();
    noise_offset = offset_;
    noise_scale = 0.002;
    PVector dir = p_0.copy();
    dir.normalize();
    p_a =dir.copy().mult(-displace_amt_);
    p_b =dir.copy().mult(displace_amt_);
  }
  void update() {
    float n = noise((position.x+position.y)*noise_scale,
    frameCount*0.002 + noise_offset)*0.8;
    PVector offset_vector = PVector.lerp(p_a, p_b, n);
    position = p_0.copy().add(offset_vector);
  }
}
class NoisySquare {
  Particle[] particles=new Particle[0];
  NoisySquare(float displace_, float offset_) {
    for (int i=0; i<sbp.getVertexCount(); i++) {
      particles=(Particle[]) append(particles,
      new Particle(sbp.getVertex(i).x,
      sbp.getVertex(i).y, displace_, offset_));
    }
  }
  void run() {
    pushMatrix();
    beginShape();
    for (int i = 0; i < particles.length; i++) {
      particles[i].update();
      PVector p = particles[i].position;
```

```
      vertex(p.x, p.y);
    }
    endShape();
    popMatrix();
  }
}
abstract class AbstractBackground {
  AbstractBackground() {
  }
  abstract void display();
}
final class SandyBackground
extends AbstractBackground {
  final PGraphics graphics;
  SandyBackground(
  float hueParameter,
  float saturationParameter,
  float minBrightness,
  float maxBrightness,  int xSize, int ySize) {
    graphics = createGraphics(xSize, ySize);
    graphics.beginDraw();
    graphics.loadPixels();
    for (int x = 0; x < xSize; x++) {
      for (int y = 0; y < ySize; y++) {
        graphics.pixels[x + y * xSize] = color(
        hueParameter,
        saturationParameter,
        random(minBrightness, maxBrightness));
      }
    }
    graphics.updatePixels();
    graphics.endDraw();
  }
  SandyBackground(
  float hueParameter,
  float saturationParameter,
  float minBrightness,
  float maxBrightness) {
    this(
    hueParameter,
    saturationParameter,
    minBrightness,
    maxBrightness, width, height);
  }
  SandyBackground() {
    this(0f, 0f, 95f, 100f);
  }
  void display() {
    image(graphics, 0f, 0f);
```

```
  }
}
void keyPressed() {
  if (key == 'r') {
    record = true;
  }
}
```

무의 코드



www.xinlin.art - /2019/16

lxl.snu.ac.kr - /2019/16

wu.pde

```
import processing.svg.*;
TTYParser parser;
Terminal terminal;
AbstractBackground currentBackground;
int BEGIN_TIME;
float SPEED = 3;
boolean record; ArrayList particles;
float mouseInfluenceSize = 15;
float mouseTearSize = 8;
float mouseInfluenceScalar = 1;
float gravity = 392;
final int curtainHeight = 2;
final int curtainWidth = 5;

final int yStart = 85;
final float restingDistances = 80;
final float stiffnesses = 1;
final float curtainTearSensitivity = 10000;
long previousTime; long currentTime;
final int fixedDeltaTime = 15;
float fixedDeltaTimeSeconds = (float)fixedDeltaTime / 1000.0;
int leftOverDeltaTime = 0;
int constraintAccuracy = 3; PFont font;
final int instructionLength = 3000;
final int instructionFade = 300;

void setup() {
  size(594, 841,P3D);
  pixelDensity(2);
  currentBackground = new SandyBackground();
  mouseInfluenceSize *= mouseInfluenceSize;
  mouseTearSize *= mouseTearSize;
  createCurtain();
```

```
  textFont(loadFont("Monaco-10.vlw"));
  textMode(SCREEN);
  parser = new TTYParser("test2");
  terminal = new Terminal(80, 24);
}
PVector wind = new PVector(0,0);
void draw(){
  if (record) {
   beginRecord(SVG, "frame1.svg");
  }
   background(#f2f2f2);
   currentTime = millis();
   long deltaTimeMS = currentTime - previousTime;
   previousTime = currentTime;
   int timeStepAmt = (int)((float)
   (deltaTimeMS + leftOverDeltaTime) / (float)fixedDeltaTime);
   timeStepAmt = min(timeStepAmt, 5);
   leftOverDeltaTime += (int)deltaTimeMS - (timeStepAmt * fixedDeltaTime);
   mouseInfluenceScalar = 1.0 / timeStepAmt;
  if(random(1)>0.99){
     wind = new PVector(random(10,100),random(0,5));
     System.out.println("windy!!!!");
  }

  for (int iteration = 1; iteration <= timeStepAmt; iteration++) {
    for (int x = 0; x < constraintAccuracy; x++) {
      for (int i = 0; i < particles.size(); i++) {
        Particle particle = (Particle) particles.get(i);
        particle.solveConstraints();
      }
    }
    for (int i = 0; i < particles.size(); i++) {
      Particle particle = (Particle) particles.get(i);
      particle.updateInteractions();
      particle.updatePhysics(fixedDeltaTimeSeconds,wind);
    }
  }
  for (int i = 0; i < particles.size(); i++) {
    Particle particle = (Particle) particles.get(i);
    particle.draw();
  }
  fill(0);
  for (int i = 0; i < parser.frames.size();i++) {
    DataFrame fr = (DataFrame)parser.frames.get(i);
    fr.update();
  }
  text(terminal.to_String(), 10, 20);
    if (record) {
    endRaw();
```

```
    record = false;
  }
 }
}
void createCurtain () {
  particles = new ArrayList();
  int midWidth = (int) (width/2 - (curtainWidth * restingDistances)/2);
  for (int y = 0; y <= curtainHeight; y++) {
    for (int x = 0; x <= curtainWidth; x++) {
      Particle particle = new Particle(new PVector
      (midWidth + x * restingDistances, y * restingDistances + yStart));
      if (x != 0)
        particle.attachTo((Particle)(particles.get(particles.size()-1)),
        restingDistances, stiffnesses);
      if (y != 0 && x!=0 && x!=curtainWidth)
        particle.attachTo((Particle)
        (particles.get((y - 1) * (curtainWidth+1) + x)),
        restingDistances, stiffnesses);
      if (x == 0  || x==curtainWidth)
        particle.pinTo(particle.position);
        particles.add(particle);
    }
  }
  for (int y = 3; y <= 140; y++) {
    for(int x=1;x<curtainWidth; x++) {
      if(y==3){
        Particle particle = new Particle(new PVector
        (midWidth + x * restingDistances, y * restingDistances + yStart));
        particle.attachTo((Particle)
        (particles.get((y - 1) * (curtainWidth+1) + x)),
        restingDistances, stiffnesses);
        particles.add(particle);
      }else{
        Particle particle = new Particle(new PVector
        (midWidth + x * restingDistances,
        3 * restingDistances + y*2 + yStart));
        particle.attachTo((Particle)(particles.get(particles.size()-4)),
        restingDistances, stiffnesses);
        particles.add(particle);
      }
    }
  }
}
void keyPressed() {
  if ((key == 'r') || (key == 'R'))
    createCurtain();
  if ((key == 'g') || (key == 'G'))
    toggleGravity();
  if (key == 'z') {
```

```
    record = true;
  }
}
void toggleGravity () {
  if (gravity != 0)
    gravity = 0;
  else
    gravity = 392;
}

float distPointToSegmentSquared (float lineX1, float lineY1,
float lineX2, float lineY2, float pointX, float pointY) {
float vx = lineX1 - pointX;
float vy = lineY1 - pointY;
float ux = lineX2 - lineX1;
float uy = lineY2 - lineY1;
float len = ux*ux + uy*uy;
float det = (-vx * ux) + (-vy * uy);
  if ((det < 0) || (det > len)) {
    ux = lineX2 - pointX;
    uy = lineY2 - pointY;
    return min(vx*vx+vy*vy, ux*ux+uy*uy);
  }
  det = ux*vy - uy*vx;
  return (det*det) / len;
}
abstract class AbstractBackground {
  AbstractBackground() {
  }

  abstract void display();
}

final class SandyBackground extends AbstractBackground {
  final PGraphics graphics;
  SandyBackground(float hueParameter, float saturationParameter,
  float minBrightness, float maxBrightness, int xSize, int ySize)
  {
    graphics = createGraphics(xSize, ySize);
    graphics.beginDraw();
    graphics.loadPixels();
    for (int x = 0; x < xSize; x++) {
      for (int y = 0; y < ySize; y++) {
        graphics.pixels[x + y * xSize] = color(hueParameter,
        saturationParameter,
        random(minBrightness, maxBrightness));
      }
    }
    graphics.updatePixels();
```

```
    graphics.endDraw();
  }
  SandyBackground(float hueParameter, float saturationParameter,
  float minBrightness, float maxBrightness) {
    this(hueParameter, saturationParameter,
    minBrightness, maxBrightness, width, height);
  }
  SandyBackground() {
    this(0f, 0f, 95f, 100f);
  }

  void display() {
    image(graphics, 0f, 0f);
  }
}
```

Link.pde

```
class Link {
  float restingDistance;
  float stiffness;
  Particle p1;
  Particle p2;
  float scalarP1;
  float scalarP2;
  boolean drawThis = true;

  Link (Particle which1, Particle which2,
  float restingDist, float stiff) {
    p1 = which1;
    p2 = which2;
    restingDistance =
    abs(p1.position.x+p1.position.y-p2.position.x-p2.position.y);
    stiffness = stiff;
    float im1 = 1 / p1.mass;
    float im2 = 1 / p2.mass;
    scalarP1 = (im1 / (im1 + im2)) * stiffness;
    scalarP2 = (im2 / (im1 + im2)) * stiffness;
  }
  void constraintSolve () {
    PVector delta = PVector.sub(p1.position, p2.position);
    float d = sqrt(delta.x * delta.x + delta.y * delta.y);
    float difference = (restingDistance - d) / d;
    if (d > curtainTearSensitivity)
      p1.removeLink(this);
      p1.position.add(PVector.mult(delta, scalarP1 * difference));
      p2.position.sub(PVector.mult(delta, scalarP2 * difference));
  }
```

```
  void draw () {
    if (drawThis)
      line(p1.position.x, p1.position.y, p2.position.x, p2.position.y);
  }
}
```

Particle.pde

```
class Particle {
  PVector lastPosition;
  PVector position;
  PVector acceleration;
  float mass = 1;
  float damping = 20;
  ArrayList links = new ArrayList();
  boolean pinned = false;
  PVector pinLocation = new PVector(0,0);

  Particle (PVector pos) {
    position = pos.get();
    lastPosition = pos.get();
    acceleration = new PVector(0,0);
  }

  void updatePhysics (float timeStep,PVector wind) {
    PVector fg = new PVector(0, mass * gravity);
    this.applyForce(fg);
    this.applyForce(wind);
    PVector velocity = PVector.sub
    (position, lastPosition);
    acceleration.sub(PVector.mult
    (velocity,damping/mass));
    PVector nextPos = PVector.add
    (PVector.add(position, velocity),
    PVector.mult(PVector.mult(acceleration, 0.5),
    timeStep * timeStep));
    lastPosition.set(position);
    position.set(nextPos);
    acceleration.set(0,0,0);
  }
  void updateInteractions () {
    if(random(1)>0.99999&&this.links.size()<=2){
      links.clear();
    }
    if (mousePressed) {
      float distanceSquared = distPointToSegmentSquared
      (pmouseX,pmouseY,mouseX,mouseY,position.x,position.y);
      if (mouseButton == LEFT) {
```

```
        if (distanceSquared < mouseInfluenceSize) {
          lastPosition = PVector.sub(position,
          new PVector((mouseX-pmouseX)*mouseInfluenceScalar,
          (mouseY-pmouseY)*mouseInfluenceScalar));
        }
      }
      else {
        if (distanceSquared < mouseTearSize)
          links.clear();
      }
    }
}

void draw () {
    stroke(0);
    if (links.size() > 0) {
      for (int i = 0; i < links.size(); i++) {
        Link currentLink = (Link) links.get(i);
        currentLink.draw();
      }
    }
    else
      point(position.x, position.y);
}
void solveConstraints () {
    for (int i = 0; i < links.size(); i++) {
      Link currentLink = (Link) links.get(i);
      currentLink.constraintSolve();
    }

    if (position.y < 1)
      position.y = 2 * (1) - position.y;
    if (position.y > height-1)
      position.y = 2 * (height - 1) - position.y;
    if (position.x > width-1)
      position.x = 2 * (width - 1) - position.x;
    if (position.x < 1)
      position.x = 2 * (1) - position.x;
    if (pinned)
      position.set(pinLocation);
}

void attachTo (Particle P, float restingDist, float stiff) {
    Link lnk = new Link(this, P, restingDist, stiff);
    links.add(lnk);
}
void removeLink (Link lnk) {
    links.remove(lnk);
}
```

```
  void applyForce (PVector f) {
    acceleration.add(PVector.div(f, mass));
  }

  void pinTo (PVector location) {
    pinned = true;
    pinLocation.set(location);
  }
}
```

Abstract

Research on the Expression of Meaning of Kinetic Chinese Characters using Computer Programming Techniques

LI XINLIN

Department of Design
College of Fine Arts
The Graduate School
Seoul National University

Scientific and technological civilizations have not only transformed all aspects of society but also brought about significant change in the field of design. Since the 1980s, young designers are brought up influenced by science and technology to the point that it can be said that they have been engulfed in the digital world since birth. These young designers have actively applied various elements of science and technology to the arts and science and technology has had a profound impact on the work of artists. Modern artists now need to go beyond the application of materials and tools available in the physical world to their work and expand into 2-dimensional, 3-dimensional, and even 4-dimensional digital art expression techniques on virtual screen spaces. As the fusion of modern art and science and technology

continue to expand, the technique of skillfully handling modern digital technologies by next generation designers has become a competitive advantage for survival in the new industrial age.

Western designers have strived to combine technology and design early on and have been able to derive systematic research methodologies as a result. On the other hand, the time period of research and adoption in the East was relatively slow compared to the digital technology advancement in the region. All Asian designers face the common question of how such theories can take root and become fruitful in the East. As an Asian artist, the author of this study attempted a method of converging computer programming technology and traditional Chinese characters with a focus on the potential of Chinese characters.

Characters or scripts are the essence of civilization and are the main subject of research in the design field. Whether it be handwritten or digital typography, character design has constantly been closely linked to the development of technology. However, conventional font design technique that used paper as the medium began to be pushed aside as it gradually failed to adapt to the dynamic(kinetic) space on the screen. Also, the method of simple changes on some screen media such as computers, TVs, and mobile phones could not fundamentally solve the kinetic design problem of moving characters. As a result, designers are carrying out various studies to converge the properties and history of a native script with digital media atop the native culture as the foundation, focusing on the script itself once again.

In this regard, the author of this study contemplated over the following problem. “In the digital age, with what method should the moving property be attributed to script? What is the directionality

that should be followed for Chinese characters that inherit the heritage of the Han(漢) dynasty?" The above content was contained in this study on "kinetic Chinese characters" composed of 6 chapters. The following is a summary of the research content and methodology.

First of all, the definitions and historical context of kinetic, kinetic art, and kinetic typography were explained and organized. Previous research and related literature were used to explore the past, present, and future of kinetic and carry out detailed analysis of kinetic related development techniques and tools. All the personal works presented in this manuscript were created using processing languages.

Next, the characteristics and development direction of Chinese characters were investigated. The design characteristics of kinetic Chinese characters were derived by bringing together the characteristics of kinetic typography. Through this study, the following were clearly determined. The process of creating kinetic Chinese characters is necessary for the adaptation of Chinese characters to the changing times and maintenance of the inherent characteristics of Chinese characters.

The analysis of artworks in this study included works that the author of this study directly participated in as well as design works of other writers. The artwork development know-how obtained through comparison between these works brought about greater clarification of the creative direction and conception.

Finally, 16 works were created based on the components of Chinese characters. The subjects of these works were all different and were made purely through computer programming and algorithms. Additionally, work appreciation essays were added to each work and this was derived from the Western culture where work appreciation

essays are occasionally used for interaction and communication with the audience and the concept of “poetry and paintings having the same source(詩畵同源)” in Asian traditional art that presents poetry and paintings together.

In this study, integration of code to type and typography revealed that the originally fixed or invariable timeline has changed. Addition of the generative technology resulted in more rich work content and changed it to become unpredictable. With this, the creative technique of kinetic typography can be applied to Chinese characters and the kinetic design integrated with the characteristics of Chinese characters holds unlimited potential.

The field of kinetic Chinese character design is currently expanding steadily, but the field is yet vague and without a clear definition. However, the everlasting Eastern culture and abundant perspective in search of beauty(審美觀) can become the basis for converging new technologies and new design so that the kinetic Chinese character design can definitely become a new trend that leads the advancement of future design.

Keywords : Kinetic, Kinetic art, Chinese Characters, Programming, Typography.
Student No. : 2017-37780
E-mail : *